

VISIT...

LANZAROTE
Caliente.COM

В.И.Ракитин, В.Е.Первушин

ПРАКТИЧЕСКОЕ РУКОВОДСТВО ПО МЕТОДАМ ВЫЧИСЛЕНИЙ

**с приложением
программ
для
персональных
компьютеров**



Москва «Высшая школа» 1998

Р е ц е н з е н т ы: кафедра вычислительной математики и программирования Московского государственного авиационного института (технического университета) (зав. кафедрой чл.-кор. РАН, проф. У.Г. Пирумов);
д-р физ.-мат. наук, проф. Э.М. Карташов

Рекомендовано

**Министерством общего и профессионального образования
Российской Федерации
для использования в учебном процессе
студентами высших технических учебных заведений**

Ракитин В.И., Первушин В.Е.

P19 Практическое руководство по методам вычислений с применением программ для персональных компьютеров: Учеб. пособие. - М.: Высш. шк., 1998. - 383 с.: ил.

ISBN 5-06-003342-2

Пособие содержит материал, предусмотренный программой по дисциплине "Вычислительная математика и программирование для ЭВМ". В каждой главе даются необходимые теоретические сведения, примеры, иллюстрирующие применение различных численных методов, и упражнения для самостоятельного решения. В приложениях приводятся блок-схемы и тексты программ на языках BASIC, PASCAL, FORTRAN.

Для студентов вузов. Может быть также полезно преподавателям, инженерам и научным работникам.

ПРЕДИСЛОВИЕ

Данное пособие написано в соответствии с программой по дисциплине "Вычислительная математика и программирование для ЭВМ", изучаемой студентами технических вузов. Значительная часть материала пособия была использована авторами при чтении курса по численным методам в Московском государственном университете инженерной экологии и факультативного курса "Компьютерное моделирование в высшей математике" в Московской государственной геологоразведочной академии.

Книга состоит из девяти глав и приложений. Эти главы охватывают следующие разделы программы: понятие линейного нормированного пространства; методы численного решения систем линейных уравнений; методы численного решения нелинейных уравнений и систем; среднеквадратичное приближение функций; интерполирование функций; численное дифференцирование и интегрирование; численное решение обыкновенных дифференциальных уравнений; численные методы поиска экстремума функций одной и нескольких переменных.

В каждой главе приводятся необходимые теоретические сведения (основные теоремы, определения, формулы, различные вычислительные методы и т.д.), а также примеры, иллюстрирующие применение описанных методов. Кроме того, имеются упражнения для самостоятельного решения и ответы к ним.

Приложения содержат блок-схемы вычислительных алгоритмов и тексты программ для рассмотренных численных методов на алгоритмических языках BASIC, PASCAL, FORTRAN и C. Тексты трех программ написаны на языке QUICKBASIC.

Основная цель пособия — помочь развитию практических навыков у читателя в применении численных методов. По мнению авторов, достижению этой цели прежде всего способствует единообразный подход к

изложению материала книги. Каждая тема содержит: вычислительный алгоритм; теоретические обоснования его применения; условия окончания вычислительного процесса; примеры, полностью или частично выполненные "вручную"; упражнения и ответы к ним; приложение, в котором рассматриваемый вычислительный алгоритм представлен в виде блок-схемы и текстов программ на четырех (иногда — на пяти) алгоритмических языках.

Авторы надеются, что овладению численными методами будет способствовать и большое количество подробно решенных примеров, а также упражнений для самостоятельной работы. Следует отметить, что часто различные вычислительные алгоритмы иллюстрируются одними и теми же примерами. Кроме того, для многих рассмотренных в книге примеров известны аналитические решения, с которыми можно сравнивать найденные численные решения. Совпадение результатов, полученных разными способами, является дополнительным, наглядным аргументом применимости того или иного численного метода.

Наконец, помощь в практическом применении численных методов окажут приложения к данной книге. В них приведены блок-схемы и тексты 95 программ (с комментариями) на используемых в учебной практике алгоритмических языках. Изложенный в приложениях материал можно применять не только при изучении численных методов, но и в качестве готовых прикладных программ, работа которых проверена в программных средах фирм BORLAND и MICROSOFT для персональных компьютеров (в программах на языке FORTRAN используется транслятор Microsoft Fortran 5.0).

Настоящее пособие предназначено для студентов высших технических учебных заведений. Оно может также оказаться полезным преподавателям, инженерам и научным работникам, использующим в своей деятельности вычислительные методы.

Авторы

ПОНЯТИЕ ЛИНЕЙНОГО НОРМИРОВАННОГО ПРОСТРАНСТВА

§1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ. ПРИМЕРЫ ЛИНЕЙНЫХ НОРМИРОВАННЫХ ПРОСТРАНСТВ

Определение. Множество L , состоящее из элементов $x, y, z, \dots$, называется линейным пространством, если на этом множестве определены две операции — сложение элементов и умножение элемента на число, удовлетворяющие следующим условиям (аксиомам):

- 1) $x + y = y + x$ $\forall x, y \in L$;
- 2) $x + (y + z) = (x + y) + z$ $\forall x, y, z \in L$;
- 3) существует "нулевой элемент" $0 \in L$ такой, что $x + 0 = x$ $\forall x \in L$;
- 4) $\forall x \in L$ существует "противоположный элемент" $-x$ такой, что $x + (-x) = 0$;
- 5) $\lambda(\mu x) = (\lambda\mu)x$ $\forall x \in L, \lambda, \mu \in R$;
- 6) $1 \cdot x = x$ $\forall x \in L$;
- 7) $(\lambda + \mu)x = \lambda x + \mu x$ $\forall x \in L, \lambda, \mu \in R$;
- 8) $\lambda(x + y) = \lambda x + \lambda y$ $\forall x, y \in L, \lambda \in R$.

Разность элементов $x - y$ определяется соотношением

$$x - y = x + (-y).$$

Определение. Линейное пространство L называется нормированным, если каждому элементу $x \in L$ поставлено в соответствие действительное число, называемое нормой элемента и обозначаемое $\|x\|$, причем удовлетворены следующие условия (аксиомы нормы):

- 1) $\|x\| \geq 0$; условие $\|x\| = 0$ эквивалентно условию $x = 0$;
- 2) $\|\lambda x\| = |\lambda| \|x\|$, $\lambda \in R$;
- 3) $\|x + y\| \leq \|x\| + \|y\|$ $\forall x, y \in L$.

Определение. Расстояние $d(x, y)$ между элементами x и y линейного нормированного пространства определяется как норма разности этих элементов, т.е.

$$d(x, y) = \|x - y\|.$$

ПРИМЕРЫ ЛИНЕЙНЫХ НОРМИРОВАННЫХ ПРОСТРАНСТВ

1. Множество действительных чисел $\mathbb{R}$ является линейным нормированным пространством, где норма — это абсолютная величина числа, а расстояние $d(x, y) = \|x - y\| = |x - y|$ ($x, y \in \mathbb{R}$).

2. Нормированные пространства n -мерных векторов

Упорядоченная совокупность n действительных чисел $x_1, x_2, \dots, x_n$ называется n -мерным вектором $\mathbf{x} \in \mathbb{R}^n$, а числа $x_1, x_2, \dots, x_n$ — его координатами.

Суммой двух n -мерных векторов $\mathbf{x}$ и $\mathbf{y}$ называется n -мерный вектор, координаты которого равны суммам соответствующих координат $x_k + y_k$ ($k=1, 2, \dots, n$) векторов $\mathbf{x}$ и $\mathbf{y}$.

Произведением n -мерного вектора $\mathbf{x}$ на число λ называется n -мерный вектор $\lambda\mathbf{x}$, координаты которого получаются из координат вектора $\mathbf{x}$ умножением их на число λ .

Будем записывать вектор $\mathbf{x}$ в виде

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

На векторах линейного пространства $\mathbb{R}^n$ можно, в частности, определить следующие три нормы (и соответственно три линейных нормированных пространства):

$$\|\mathbf{x}\|_m = \max\{|x_1|, |x_2|, \dots, |x_n|\} \quad (m\text{-норма или кубическая норма});$$

$$\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i| \quad (1\text{-норма или октаэдрическая норма});$$

$$\|\mathbf{x}\|_k = \sqrt{\sum_{i=1}^n x_i^2} \quad (k\text{-норма или сферическая норма}).$$

Пример 1. Вычислить расстояние между векторами $\mathbf{x}$ и $\mathbf{y}$ для каждой из трех введенных норм, если

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ -3 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}.$$

Решение. Найдем вектор разности

$$\mathbf{x} - \mathbf{y} = \begin{bmatrix} 0 \\ 3 \\ -4 \end{bmatrix}.$$

Согласно определению расстояния между векторами как нормы их разности, получим

$$d(\mathbf{x}, \mathbf{y})_m = \|\mathbf{x} - \mathbf{y}\|_m = \max \{0, 3, 4\} = 4;$$

$$d(\mathbf{x}, \mathbf{y})_l = \|\mathbf{x} - \mathbf{y}\|_l = 0 + 3 + 4 = 7;$$

$$d(\mathbf{x}, \mathbf{y})_k = \|\mathbf{x} - \mathbf{y}\|_k = \sqrt{0^2 + 3^2 + 4^2} = 5.$$

Пример 2. На плоскости R^2 найти множества точек, удовлетворяющих условию $\|\mathbf{x}\| = 1$. Решение получить в пространстве R^2 для трех норм: а) m -нормы; б) l -нормы; в) k -нормы.

Решение. По определению, сфера (в частности, окружность) — это множество точек, находящихся на заданном расстоянии от некоторой точки, называемой центром сферы. Условие $\|\mathbf{x}\| = 1$ определяет сферу единичного радиуса с центром в начале координат. Геометрический образ сферы зависит от рассматриваемого линейного нормированного пространства. В данном примере получим три различных геометрических образа формально определенной сферы (рис. 1–3).

На плоскости Ox_1x_2 точке (x_1, x_2) соответствует вектор $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in R^2$.

а) В пространстве R^2 с m -нормой условие $\|\mathbf{x}\|_m = 1$ эквивалентно соотношению $\max \{|x_1|, |x_2|\} = 1$ или объединению двух множеств точек на плоскости Ox_1x_2 : $\{(x_1, x_2) : |x_1| = 1, |x_2| \leq 1\}$ и $\{(x_1, x_2) : |x_1| \leq 1, |x_2| = 1\}$. Это множество является границей квадрата (рис. 1).

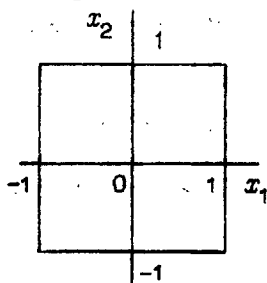


Рис. 1

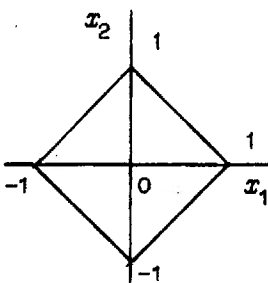


Рис. 2

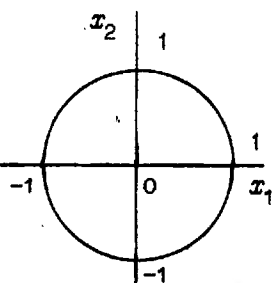


Рис. 3

б) В пространстве R^2 с 1-нормой равенство $\|x\|_1 = |x_1| + |x_2| = 1$ определяет множество точек, образующих границу квадрата (рис.2). Действительно, условие $|x_1| + |x_2| = 1$ эквивалентно соотношениям

$$\begin{cases} x_1 + x_2 = 1 & \text{при } x_1 \geq 0, x_2 \geq 0; \\ x_1 - x_2 = 1 & \text{при } x_1 \geq 0, x_2 < 0; \\ -x_1 + x_2 = 1 & \text{при } x_1 < 0, x_2 \geq 0; \\ -x_1 - x_2 = 1 & \text{при } x_1 < 0, x_2 < 0, \end{cases}$$

которые позволяют выполнить чертеж последовательно в каждой четверти плоскости Ox_1x_2 .

в) По k -норме условие $\|x\|_k = \sqrt{x_1^2 + x_2^2} = 1$ определяет удаленные на единицу от начала координат точки, лежащие на окружности (рис.3). ■

Можно убедиться, что в пространстве R^3 с m -, l -, k -нормами условие $\|x\|_m = \|x\|_l = \|x\|_k = 1$ определяет множества точек, лежащих соответственно на гранях куба, октаэдра и на поверхности сферы. Последнее мотивирует названия введенных норм.

3. Нормированные пространства матриц размера n на m

Определение. Прямоугольная таблица чисел из n строк и m столбцов называется **матрицей** размера n на m .

Будем обозначать матрицу так:

$$A = (a_{ij})_{n \times m} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}$$

Числа a_{ij} ($i=1,2,\dots,n$; $j=1,2,\dots,m$) называются **элементами** матрицы. Нижние индексы i и j элемента a_{ij} указывают номера строки и столбца, на пересечении которых находится этот элемент в таблице. Матрица A называется **квадратной** порядка n , если $n=m$.

Матрицы одинакового размера образуют линейное пространство, если для любой пары матриц $A=(a_{ij})$ и $B=(b_{ij})$ определена матрица

суммы $A+B$, составленная из элементов $(a_{ij} + b_{ij})$, а умножение матрицы на число определено как умножение каждого элемента матрицы на это число ($i=1, 2, \dots, n$; $j=1, 2, \dots, m$).

Для матриц $A=(a_{ij})_{n \times m}$ определим следующие три нормы:

$$\|A\|_m = \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^m |a_{ij}| \right\} \quad (m\text{-норма});$$

$$\|A\|_l = \max_{1 \leq j \leq m} \left\{ \sum_{i=1}^n |a_{ij}| \right\} \quad (l\text{-норма});$$

$$\|A\|_k = \sqrt{\sum_{i=1}^n \sum_{j=1}^m a_{ij}^2} \quad (k\text{-норма}).$$

При $m=1$ матрица A представляет собой n -мерный вектор. Ранее введенные нормы для векторов совпадают с нормами, определенными в частном случае для матриц размера $n \times 1$.

Пример 3. Вычислить нормы матрицы

$$A = \begin{bmatrix} 1 & -1 & 2 \\ 1 & 3 & -1 \\ 2 & 0 & -1 \\ 1 & 1 & -1 \end{bmatrix}.$$

Решение. Используя определения норм, получим

$$\|A\|_m = \max \{1+1+2, 1+3+1, 2+0+1, 1+1+1\} = \max \{4, 5, 3, 3\} = 5;$$

$$\|A\|_l = \max \{1+1+2+1, 1+3+0+1, 2+1+1+1\} = \max \{5, 5, 5\} = 5;$$

$$\|A\|_k = \sqrt{(1^2+1^2+2^2)+(1^2+3^2+1^2)+(2^2+0^2+1^2)+(1^2+1^2+1^2)} = 5. \blacksquare$$

В каждой точке некоторой области пространства R^n могут быть определены n функций, непрерывных вместе со своими частными производными. Тогда в каждой точке можно определить функциональную матрицу порядка n .

Определение. Матрицей Якоби системы функций $\{f_1(x_1, x_2, \dots, x_n), f_2(x_1, x_2, \dots, x_n), \dots, f_n(x_1, x_2, \dots, x_n)\}$ называется функциональная квадратная матрица порядка n следующего вида:

$$J(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}.$$

Нормы матрицы Якоби в точке $\mathbf{x}$ вычисляются по формулам

$$\|J(\mathbf{x})\|_m = \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^n \left| \frac{\partial f_i}{\partial x_j} \right| \right\} \quad (m\text{-норма});$$

$$\|J(\mathbf{x})\|_l = \max_{1 \leq j \leq n} \left\{ \sum_{i=1}^n \left| \frac{\partial f_i}{\partial x_j} \right| \right\} \quad (l\text{-норма});$$

$$\|J(\mathbf{x})\|_k = \sqrt{\sum_{i=1}^n \sum_{j=1}^n \left(\frac{\partial f_i}{\partial x_j} \right)^2} \quad (k\text{-норма}).$$

Пример 4. Дана система функций $\left\{ \frac{1}{4} (x_1 + x_2^2); \frac{1}{2} x_1^2 + 1 \right\}$.

Найти множество $\mathbf{x} \in \mathbb{R}^2$, на котором выполняется неравенство

$$\|J(\mathbf{x})\|_m < 1.$$

Решение. Здесь $f_1(x_1, x_2) = \frac{1}{4} (x_1 + x_2^2)$; $f_2(x_1, x_2) = \frac{1}{2} x_1^2 + 1$.

Из определений матрицы Якоби и m -нормы матрицы следует

$$J(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} \frac{1}{4} & x_2 \\ x_1 & 0 \end{bmatrix};$$

$$\|J(\mathbf{x})\|_m = \max \left\{ \frac{1}{4} + \frac{|x_2|}{2}; |x_1| \right\};$$

$$\|J(\mathbf{x})\|_m < 1 \iff \begin{cases} \frac{1}{4} + \frac{|x_2|}{2} < 1, \\ |x_1| < 1 \end{cases} \iff \begin{cases} |x_1| < 1, \\ |x_2| < \frac{3}{2}. \end{cases}$$

Заданное неравенство выполняется в точках открытого прямоугольника $\{(x_1, x_2): |x_1| < 1, |x_2| < 3/2\}$ (рис. 4). ■

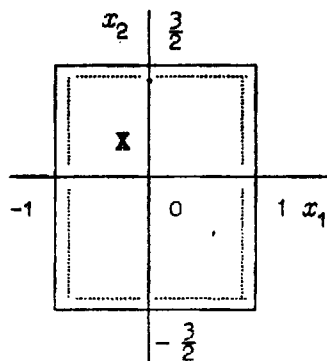


Рис. 4

4. Нормированные пространства на множестве функций, непрерывных на отрезке $[a, b]$

Можно показать, что множество непрерывных функций на отрезке $[a, b]$ относительно операций сложения $x(t) + y(t)$ и умножения на число $\lambda x(t)$ образует линейное пространство $C_{[a, b]}$ ($t \in [a, b]$; $\lambda \in \mathbb{R}$; $x(t), y(t) \in C_{[a, b]}$).

Норма $\|x\|_C$ называется **равномерной** на $C_{[a, b]}$ или **C -нормой**, если

$$\|x\|_C = \max_{a \leq t \leq b} |x(t)|.$$

Норма $\|x\|_{L^2}$ называется **квадратичной** на $C_{[a, b]}$, если

$$\|x\|_{L^2} = \sqrt{\int_a^b x^2(t) dt}.$$

Расстояния между функциями $x(t), y(t) \in C_{[a, b]}$ по равномерной и квадратичной нормам вычисляются соответственно по формулам

$$d(x, y)_C = \|x - y\|_C = \max_{a \leq t \leq b} |x(t) - y(t)|,$$

$$d(x, y)_{L^2} = \|x - y\|_{L^2} = \sqrt{\int_a^b (x(t) - y(t))^2 dt}.$$

Пример 5. Вычислить равномерную и квадратичную нормы функций $x(t)$ и $y(t)$ и определить расстояния $d(x, y)_C$ и $d(x, y)_{L^2}$, если $x(t) = \sin t$, $y(t) = \cos t$ ($t \in [0, \pi]$).

Решение. Используя приведенные определения, получим

$$\|\sin t\|_C = \max_{0 \leq t \leq \pi} |\sin t| = 1, \quad \|\cos t\|_C = \max_{0 \leq t \leq \pi} |\cos t| = 1,$$

$$d(\sin t, \cos t)_C = \|\sin t - \cos t\|_C = \max_{0 \leq t \leq \pi} |\sin t - \cos t| =$$

$$= \max_{0 \leq t \leq \pi} |\sin t - \sin(\frac{\pi}{2} - t)| = 2 \cos \frac{\pi}{4} \max_{0 \leq t \leq \pi} |\sin(t - \frac{\pi}{4})| = \sqrt{2},$$

$$\|\sin t\|_{L^2} = \sqrt{\int_0^{\pi} \sin^2 t dt} = \sqrt{\int_0^{\pi} \frac{1 - \cos 2t}{2} dt} = \frac{1}{\sqrt{2}} \sqrt{\int_0^{\pi} dt - \frac{1}{2} \int_0^{\pi} \cos 2t d(2t)} =$$

$$= \frac{1}{\sqrt{2}} \sqrt{t \Big|_0^{\pi} - \frac{1}{2} \sin 2t \Big|_0^{\pi}} = \sqrt{\frac{\pi}{2}},$$

$$\|\cos t\|_{L^2} = \sqrt{\int_0^{\pi} \cos^2 t dt} = \sqrt{\int_0^{\pi} \frac{1 + \cos 2t}{2} dt} = \sqrt{\frac{\pi}{2}},$$

$$d(\sin t, \cos t)_{L^2} = \|\sin t - \cos t\|_{L^2} =$$

$$= \sqrt{\int_0^{\pi} (\sin t - \cos t)^2 dt} = \sqrt{\int_0^{\pi} (1 - 2 \sin t \cos t) dt} =$$

$$= \sqrt{t \Big|_0^{\pi} - 2 \int_0^{\pi} \sin t d(\sin t)} = \sqrt{\pi - \sin^2 t \Big|_0^{\pi}} = \sqrt{\pi}.$$

Заметим, что для заданной функции $x(t)$ и заданного числа $\varepsilon > 0$ неравенство $d(x, y)_C < \varepsilon$ означает, в частности, что график функции $y(t)$ расположен в полосе между графиками функций $x(t) - \varepsilon$ и $x(t) + \varepsilon$ (рис. 5). Это вытекает из равносильности неравенств

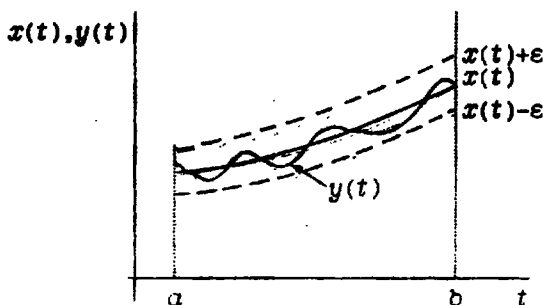


Рис. 5

$$\begin{aligned}
 d(x, y)_C < \varepsilon &\iff \|x - y\| < \varepsilon \iff \\
 &\iff |x(t) - y(t)| < \varepsilon \quad \forall t \in [a, b] \iff \\
 &\iff x(t) - \varepsilon < y(t) < x(t) + \varepsilon \quad \forall t \in [a, b].
 \end{aligned}$$

Упражнения

1. Даны векторы $a = \begin{bmatrix} 5 \\ -6 \end{bmatrix}$, $b = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$, $c = \begin{bmatrix} 4 \\ 8 \\ -10 \\ 0 \end{bmatrix}$, $d = \begin{bmatrix} 2 \\ 0 \\ 1 \\ -3 \end{bmatrix}$. Вычислить их m -, l - и k -нормы и найти расстояния $d(a, b)$ и $d(c, d)$ для каждой из этих норм.

2. Даны два вектора $a = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$ и $b = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$. На координатной плоскости Ox_1x_2 указать множество точек (x_1, x_2) , для которых:

а) $\|x - a\|_m \leq 2$; б) $\|x - b\|_k > 1$; в) $\|x + b\|_l = 2$;
 г) $\|x - (a - 2b)\|_m \leq 3$; д) $\|x - a\|_k + \|x - b\|_k = 4$.

3. На плоскости R^2 указать множество точек X , для которых выполняется неравенство $\|J(X)\|_m < 1$, где $J(X)$ — матрица Якоби следующих систем функций:

а) $f_1(x_1, x_2) = x_1^2 + \frac{1}{2} x_2$, $f_2(x_1, x_2) = \frac{3}{4} x_1 + \frac{1}{3} x_2^3$;

б) $f_1(x_1, x_2) = \ln x_2$, $f_2(x_1, x_2) = \frac{1}{4} x_1^2 - \frac{1}{2} x_2$;

в) $f_1(x_1, x_2) = \frac{1}{2} x_2^2 - x_2$, $f_2(x_1, x_2) = \frac{1}{2} \cos x_1$;

$$\Gamma) f_1(x_1, x_2) = \frac{1}{2} x_1^2 + \frac{1}{2} x_2, \quad f_2(x_1, x_2) = x_2^2 + \frac{1}{2} x_2;$$

$$\Delta) f_1(x_1, x_2) = \cos x_2, \quad f_2(x_1, x_2) = \sin x_1.$$

4. Вычислить равномерные и квадратичные нормы функций $x(t)$ и $y(t)$ и определить расстояния $d(x, y)_C$, $d(x, y)_{L^2}$, если:

$$a) x(t) = 1 - t, \quad y(t) = t^2 - t, \quad t \in [0, 1];$$

$$б) x(t) = e^t, \quad y(t) = 1 - t, \quad t \in [0, 1];$$

$$в) x(t) = \ln t, \quad y(t) = t, \quad t \in [1, 2];$$

$$\Gamma) x(t) = \sin t, \quad y(t) = 1 - \frac{1}{\pi} t, \quad t \in [0, \pi].$$

§2 СХОДИМОСТЬ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

В ЛИНЕЙНЫХ НОРМИРОВАННЫХ ПРОСТРАНСТВАХ

Определение. Последовательность элементов $\{x^{(k)}\}$ ($k=1, 2, \dots$) линейного нормированного пространства называется **сходящейся** к элементу a этого пространства, если расстояние $d(x^{(k)}, a) \rightarrow 0$ при $k \rightarrow \infty$. При этом элемент a называют **пределом последовательности** $\{x^{(k)}\}$ ($k=1, 2, \dots$) и пишут $a = \lim_{k \rightarrow \infty} x^{(k)}$.

1.0. Сходимость последовательностей n -мерных векторов и матриц

Пусть в пространстве R^n дана последовательность векторов $\{x^{(k)}\}$ ($k=1, 2, \dots$):

$$x^{(1)} = \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_n^{(1)} \end{bmatrix}, \quad x^{(2)} = \begin{bmatrix} x_1^{(2)} \\ x_2^{(2)} \\ \vdots \\ x_n^{(2)} \end{bmatrix}, \dots, \quad x^{(k)} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \\ \vdots \\ x_n^{(k)} \end{bmatrix}, \dots \quad (1)$$

и некоторый вектор $a = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$; тогда справедлива следующая

теорема.

Теорема 1. Последовательность векторов (1) сходится (по $\|\cdot\|$, l -, k - нормам) при $k \rightarrow \infty$ к вектору $\mathbf{a}$ тогда и только тогда, когда существуют пределы числовых последовательностей координат векторов $\mathbf{x}^{(k)}$:

$$\alpha_1 = \lim_{k \rightarrow \infty} x_1^{(k)}, \quad \alpha_2 = \lim_{k \rightarrow \infty} x_2^{(k)}, \dots, \quad \alpha_n = \lim_{k \rightarrow \infty} x_n^{(k)},$$

т.е. сходимость последовательности векторов (1) равносильна покоординатной сходимости этих векторов.

Пример 1. Показать, что последовательность векторов

$$\mathbf{x}^{(k)} = \begin{bmatrix} 1 - (1/k) \\ 1 + (1/k) \\ -2/\sqrt{k} \\ 3 + (1/k) \end{bmatrix} \quad \text{сходится к вектору } \mathbf{a} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 3 \end{bmatrix}.$$

Для любого $\varepsilon > 0$ найти номер, начиная с которого для всех членов последовательности векторов $\mathbf{x}^{(k)}$ выполняется неравенство

$$d(\mathbf{x}^{(k)}, \mathbf{a})_m = \|\mathbf{x}^{(k)} - \mathbf{a}\|_m < \varepsilon.$$

Решение. На основании теоремы 1 сходимость последовательности векторов $\mathbf{x}^{(k)}$ к вектору $\mathbf{a}$ следует из того, что существуют пределы

$$\alpha_1 = \lim_{k \rightarrow \infty} x_1^{(k)} = \lim_{k \rightarrow \infty} \left[1 - \frac{1}{k} \right] = 1, \quad \alpha_2 = \lim_{k \rightarrow \infty} x_2^{(k)} = \lim_{k \rightarrow \infty} \left[1 + \frac{1}{k} \right] = 1,$$

$$\alpha_3 = \lim_{k \rightarrow \infty} x_3^{(k)} = \lim_{k \rightarrow \infty} \left[-\frac{2}{\sqrt{k}} \right] = 0, \quad \alpha_4 = \lim_{k \rightarrow \infty} x_4^{(k)} = \lim_{k \rightarrow \infty} \left[3 + \frac{1}{k} \right] = 3.$$

Кроме того, $\mathbf{x}^{(k)} - \mathbf{a} = \begin{bmatrix} -1/k \\ 1/k \\ -2/\sqrt{k} \\ 1/k \end{bmatrix}$ и $d(\mathbf{x}^{(k)}, \mathbf{a})_m = \|\mathbf{x}^{(k)} - \mathbf{a}\|_m =$

$$= \max \left\{ \frac{1}{k}, \frac{1}{k}, \frac{2}{\sqrt{k}}, \frac{1}{k} \right\} = \frac{2}{\sqrt{k}} < \varepsilon; \quad \text{тогда для любого } \varepsilon > 0 \text{ рас-}$$

стояние $d(\mathbf{x}^{(k)}, \mathbf{a}) < \varepsilon$, начиная с номера $k > \frac{4}{\varepsilon^2}$ (в частности, если $\varepsilon = 10^{-2}$, то $k > 40000$). ■

В линейном нормированном пространстве матриц размера $n \times m$ справедлива теорема, аналогичная теореме 1.

Теорема 2. Пусть в пространстве матриц $M_{n \times m}$ задана последовательность матриц $\{A^{(k)}\}$:

$$A^{(1)} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1m}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \dots & a_{2m}^{(1)} \\ \dots & \dots & \dots & \dots \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \dots & a_{nm}^{(1)} \end{bmatrix}, \dots, A^{(k)} = \begin{bmatrix} a_{11}^{(k)} & a_{12}^{(k)} & \dots & a_{1m}^{(k)} \\ a_{21}^{(k)} & a_{22}^{(k)} & \dots & a_{2m}^{(k)} \\ \dots & \dots & \dots & \dots \\ a_{n1}^{(k)} & a_{n2}^{(k)} & \dots & a_{nm}^{(k)} \end{bmatrix}, \dots \quad (2)$$

Последовательность (2) сходится (по m -, l - или k -норме) к матрице

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix}$$

тогда и только тогда, когда последовательности элементов $a_{ij}^{(k)}$ матриц $A^{(k)}$ сходятся к соответствующим элементам a_{ij} матрицы A :

$$a_{11} = \lim_{k \rightarrow \infty} a_{11}^{(k)}, a_{12} = \lim_{k \rightarrow \infty} a_{12}^{(k)}, \dots, a_{ij} = \lim_{k \rightarrow \infty} a_{ij}^{(k)}, \dots, a_{nm} = \lim_{k \rightarrow \infty} a_{nm}^{(k)}.$$

2°. Сходимость последовательностей непрерывных функций

Пусть на отрезке $[a, b]$ определена последовательность непрерывных функций:

$$x^{(1)}(t), x^{(2)}(t), \dots, x^{(k)}(t), \dots \quad (3)$$

Определение. Говорят, что последовательность функций $\{x^{(k)}(t)\}$ ($k=1, 2, \dots$) сходится в точке $t_0 \in [a, b]$, если существует предел числовой последовательности $\{x^{(k)}(t_0)\}$.

Определение. Последовательность (3) сходится поточечно к функции $x(t)$ на отрезке $[a, b]$, если она сходится в каждой точке этого отрезка к значению функции $x(t)$. Иначе, если для $\forall \varepsilon > 0 \exists K(\varepsilon, t)$ такое, что $\forall k > K(\varepsilon, t)$ выполняется неравенство

$$|x^{(k)}(t) - x(t)| < \varepsilon \quad (t \in [a, b]).$$

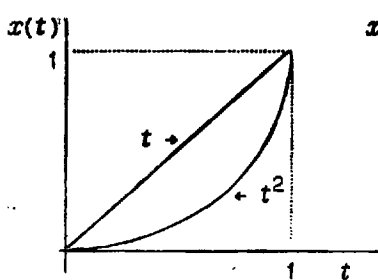


Рис.6

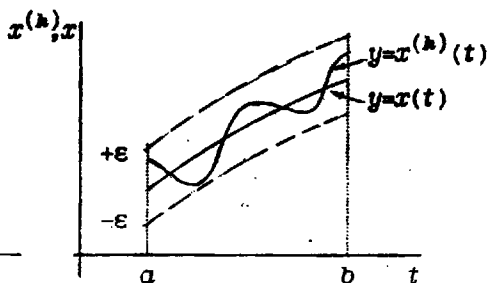


Рис.7

Пример 2. Последовательность степенных функций $t, t^2, \dots, t^k, \dots$ сходится поточечно на отрезке $[0, 1]$ к разрывной функции (рис.6)

$$x(t) = \begin{cases} 0 & \text{при } 0 \leq t < 1; \\ 1 & \text{при } t = 1. \end{cases}$$

Определение. Говорят, что последовательность функций $\{x^{(k)}(t)\}$ сходится равномерно к функции $x(t)$ на отрезке $[a, b]$, если расстояние между общим членом последовательности $x^{(k)}(t)$ и функцией $x(t)$ стремится к нулю при $k \rightarrow \infty$ по равномерной норме, т.е.

$$\lim_{k \rightarrow \infty} d(x^{(k)}, x)_C = \lim_{k \rightarrow \infty} \max_{a \leq t \leq b} |x^{(k)}(t) - x(t)| = 0.$$

Иначе говоря, для $\forall \varepsilon > 0$ $\exists K(\varepsilon)$ такое, что для $\forall k > K(\varepsilon)$ и для $\forall t \in [a, b]$ выполняется неравенство $|x^{(k)}(t) - x(t)| < \varepsilon$ или, начиная с некоторого числа $K(\varepsilon)$, выполняются соотношения $x(t) - \varepsilon < x^{(k)}(t) < x(t) + \varepsilon$.

Геометрически это означает, что графики функций $x^{(k)}(t)$ на отрезке $[a, b]$ находятся внутри полосы шириной 2ε , центром которой является график функции $x(t)$ (рис.7).

Пример 3. Последовательность функций $x^{(k)}(t) = \frac{kt}{1+k+t}$ ($k=1, 2, \dots$) определена на отрезке $[0, 1]$. Она сходится поточечно к функции $x(t)=t$ на отрезке $[0, 1]$, так как

$$\lim_{k \rightarrow \infty} x^{(k)}(t) = \lim_{k \rightarrow \infty} \frac{kt}{1+k+t} = \lim_{k \rightarrow \infty} \frac{t}{\frac{1}{k} + 1 + \frac{t}{k}} = t.$$

Эта последовательность сходится равномерно к той же функции

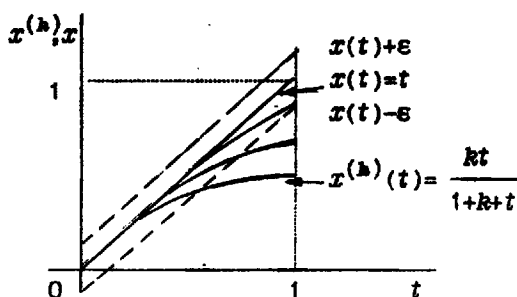


Рис. 8

$x(t) = t$ на отрезке $[0,1]$. Действительно,

$$\begin{aligned} d(x^{(k)}, x)_C &= \|x^{(k)} - x\|_C = \max_{0 \leq t \leq 1} \left| \frac{kt}{1+k+t} - t \right| = \max_{0 \leq t \leq 1} \frac{|-t-t^2|}{1+k+t} = \\ &= \max_{0 \leq t \leq 1} \frac{t + t^2}{1+k+t} \leq \frac{2}{1+k} \rightarrow 0 \text{ при } k \rightarrow \infty. \end{aligned}$$

На рис. 8 показано взаимное расположение предельной функции $x(t)=t$ и некоторых членов последовательности $x^{(k)}(t)$ на $[0,1]$.

Определение. Говорят, что последовательность функций $\{x^{(k)}(t)\}$ ($k=0,1,\dots$) сходится в среднем квадратичном к функции $x(t)$ на отрезке $[a,b]$, если расстояние между общим членом последовательности $x^{(k)}(t)$ и функцией $x(t)$ стремится к нулю при $k \rightarrow \infty$ по квадратичной норме, т.е.

$$\lim_{k \rightarrow \infty} d(x^{(k)}, x)_{L^2} = \lim_{k \rightarrow \infty} \sqrt{\int_a^b (x^{(k)}(t) - x(t))^2 dt} = 0.$$

Заметим, что из равномерной сходимости последовательности функций $\{x^{(k)}(t)\}$ к функции $x(t)$ на отрезке $[a,b]$ следует ее сходимость в среднем квадратичном к функции $x(t)$ на $[a,b]$:

$$\text{из неравенства } \int_a^b (x^{(k)}(t) - x(t))^2 dt \leq \max_{a \leq t \leq b} (x^{(k)}(t) - x(t))^2 (b-a)$$

видно, что если $d(x^{(k)}, x)_C \rightarrow 0$, то и $d(x^{(k)}, x)_{L^2} \rightarrow 0$. Обратное утверждение неверно.

Пример 4. Показать, что последовательность функций

$x^{(k)}(t) = \frac{kt}{1+kt}$ ($k=1, 2, \dots$) сходится к функции $x(t) \equiv 1$ на отрезке

$[0, 1]$ в среднем квадратичном, но не сходится к ней равномерно.

Решение. Вычислим расстояния между $x^{(k)}(t)$ и $x(t)$ по квадратичной и равномерной нормам:

$$\begin{aligned} d(x^{(k)}, x)_{L^2} &= \sqrt{\int_0^1 \left[\frac{kt}{1+kt} - 1 \right]^2 dt} = \sqrt{\int_0^1 \frac{dt}{(1+kt)^2}} = \sqrt{\left[-\frac{1}{k(1+kt)} \right]_0^1} = \\ &= \sqrt{-\left[\frac{1}{k(1+k)} - \frac{1}{k} \right]} = \frac{1}{\sqrt{1+k}} \rightarrow 0 \text{ при } k \rightarrow \infty \text{ (сходится в среднем);} \end{aligned}$$

$$d(x^{(k)}, x)_C = \max_{0 \leq t \leq 1} \left| \frac{kt}{1+kt} - 1 \right| = \max_{0 \leq t \leq 1} \frac{1}{|1+kt|} = 1 \neq 0 \text{ при } k \rightarrow \infty$$

(последовательность не сходится равномерно к функции $x(t) \equiv 1$).

На рис. 9 изображены графики функций $x(t) \equiv 1$ и $x^{(k)}(t) = kt/(1+kt)$. Очевидно, что в окрестности точки $t=0$ для любого номера k функции-члены последовательности не приближаются к предельной функции на $[0, 1]$ (не входят в ε -окрестность по равномерной норме). ■

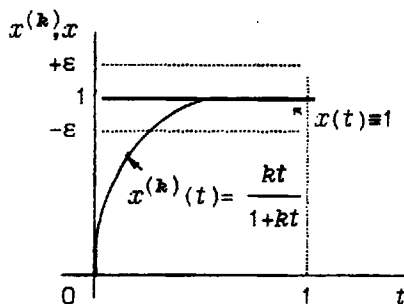


Рис. 9

Упражнения

1. Найти пределы последовательностей векторов и матриц при $k \rightarrow \infty$:

$$\text{а) } \begin{bmatrix} \frac{k+1}{k} \\ \frac{(k-1)^2}{2k^2} \end{bmatrix}; \quad \text{б) } \begin{bmatrix} \frac{\sqrt[3]{8k^3+k}}{k} & \frac{10 \ln k}{k} \\ \left(1 + \frac{1}{k}\right)^3 & \frac{3^{k+1}}{5^k} \end{bmatrix}; \quad \text{в) } \begin{bmatrix} \frac{k^3+1}{k^3+2k} \\ -2 + \frac{1}{k} \\ \frac{\sqrt{k-1}}{k} \end{bmatrix}.$$

2. Исследовать на равномерную сходимость последовательности функций $x^{(k)}(t)$:

$$\text{а) } \frac{\sin kt}{k} \quad (0 \leq t \leq 2\pi; \quad k=1, 2, \dots); \quad \text{б) } \frac{kt}{1+k^2 t^2} \quad (0 \leq t \leq 1; \quad k=1, 2, \dots);$$

$$\text{в) } \sqrt{t^2 + \frac{1}{k^2}} \quad (-\infty < t < \infty; \quad k=1, 2, \dots); \quad \text{г) } \frac{1}{\sqrt{1+kt}} \quad (0 \leq t \leq 1; \quad k=1, 2, \dots);$$

$$\text{д) } \frac{t}{k} \quad (0 \leq t \leq 1; \quad k=1, 2, \dots); \quad \text{е) } \frac{t}{1+k^2 t} \quad (0 \leq t \leq 1; \quad k=1, 2, \dots).$$

3. Найти пределы последовательностей в среднем квадратичном:

$$\text{а) } t^k \quad (0 \leq t \leq 1; \quad k=1, 2, \dots); \quad \text{б) } \frac{1}{\sqrt{1+kt}} \quad (0 \leq t \leq 1; \quad k=1, 2, \dots).$$

4. Показать, что последовательность функций $x^{(k)}(t)$ сходится в среднем квадратичном к предельной функции $x(t)$:

$$\text{а) } x^{(k)}(t) = 1 - t^k \quad (0 \leq t \leq 1; \quad k=1, 2, \dots), \quad x(t) = 1 \quad (0 \leq t \leq 1);$$

$$\text{б) } x^{(k)}(t) = 1 - t^{2k} \quad (-1 \leq t \leq 1; \quad k=1, 2, \dots), \quad x(t) = 1 \quad (-1 \leq t \leq 1);$$

$$\text{в) } x^{(k)}(t) = \begin{cases} -1, & -1 \leq t \leq 1/k; \\ kt, & -1/k \leq t \leq 1/k; \\ 1, & 1/k \leq t \leq 1; \end{cases} \quad x(t) = \begin{cases} -1, & -1 \leq t < 0; \\ 1, & 0 < t \leq 1; \end{cases} \\ (k=1, 2, \dots).$$

**МЕТОДЫ ЧИСЛЕННОГО РЕШЕНИЯ СИСТЕМ
ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ**

§1. МЕТОД ГАУССА

Метод Гаусса (метод исключения неизвестных) сначала поясним на примере. Пусть требуется решить систему уравнений

$$\begin{cases} 5x_1 - x_2 + 2x_3 = 8, \\ x_1 + 4x_2 - x_3 = -4, \\ x_1 + x_2 + 4x_3 = 4. \end{cases} \quad (1)$$

Исключим сначала неизвестное x_1 из второго и третьего уравнений системы (1), используя первое уравнение.

Уравнение, с помощью которого преобразуют остальные уравнения, иногда называют **разрешающим уравнением**, а коэффициент этого уравнения при неизвестном, исключаемом из остальных уравнений, - **разрешающим (или главным) элементом**. В данном примере первое уравнение (или первая строка) - разрешающее, а коэффициент 5 при x_1 в этом уравнении - разрешающий элемент.

Разделим первое (разрешающее) уравнение на 5 и вычтем преобразованное первое уравнение из второго и третьего уравнений системы (1). В результате получим систему, в которой неизвестное x_1 исключено из второго и третьего уравнений:

$$\begin{cases} x_1 - \frac{1}{5}x_2 + \frac{2}{5}x_3 = \frac{8}{5}, \\ \frac{21}{5}x_2 - \frac{7}{5}x_3 = -\frac{28}{5}, \\ \frac{6}{5}x_2 + \frac{18}{5}x_3 = \frac{12}{5}. \end{cases}$$

Теперь второе уравнение будет разрешающим с разрешающим элементом $\frac{21}{5}$. Разделим второе уравнение на $\frac{21}{5}$ и вычтем преобразованное второе уравнение, умноженное на $\frac{6}{5}$, из третьего уравнения, исключая тем самым неизвестное x_2 в последнем уравнении. Получим систему

$$\begin{cases} x_1 - \frac{1}{5}x_2 + \frac{2}{5}x_3 = \frac{8}{5}, \\ x_2 - \frac{1}{3}x_3 = -\frac{4}{3}, \\ x_3 = 1. \end{cases}$$

Проведенную последовательность преобразований данной системы называют **прямым ходом** в методе Гаусса. **Обратный ход** - это последовательное исключение неизвестных x_3 и x_2 из второго и первого уравнений.

Умножим третье уравнение на $(-\frac{1}{3})$ и вычтем его из второго, затем умножим это же третье уравнение на $\frac{2}{5}$ и вычтем его из первого. Получим

$$\begin{cases} x_1 - \frac{1}{5}x_2 & = \frac{6}{5}, \\ x_2 & = -1, \\ x_3 & = 1. \end{cases}$$

Далее умножим второе уравнение на $(-\frac{1}{5})$ и, используя это уравнение, исключим неизвестное x_2 из первого уравнения. Окончательно имеем

$$\begin{cases} x_1 & = 1, \\ x_2 & = -1, \\ x_3 & = 1. \end{cases} \quad (2)$$

Решение последней системы очевидно.

Пусть дана система из n линейных алгебраических уравнений с n неизвестными:

[illegible]

Решением системы (3) называется упорядоченное множество чисел

Затем исключаем неизвестное x_{n-2} из уравнений с номером j ($j = n-2, \dots, 1$) и т. д. Приходим к системе, аналогичной (2). Вычисления заканчиваются решением системы, имеющим вид

$$\left\{ \begin{array}{lcl} x_1 & \dots & = b_1^{(1, n-1)}, \\ & \dots & \dots \\ & x_{n-j} & = b_{n-j}^{(n-j, j)}, \\ & \dots & \dots \\ & & \dots \\ & x_{n-2} & = b_{n-2}^{(n-2, 2)}, \\ & & \dots \\ & x_{n-1} & = b_{n-1}^{(n-1, 1)}, \\ & & \dots \\ & x_n & = b_n^{(n)}. \end{array} \right.$$

Для уменьшения погрешности вычислений существуют различные модификации метода Гаусса. Одна из модификаций метода Гаусса с **выбором максимального элемента по столбцу** состоит в следующем.

В начале первого шага прямого хода среди коэффициентов a_{i1} ($i=1, 2, \dots, n$) при неизвестном x_1 находят наибольший по модулю. Предположим, что это a_{j1} . После этого в исходной системе (3) можно произвести перестановку: первое уравнение можно поставить на место j -го, а j -е на место первого. Далее вычисления проводят в описанной ранее последовательности.

В начале второго шага прямого хода максимальный по модулю элемент выбирают среди коэффициентов a_{i2} ($i = 2, 3, \dots, n$) при неизвестном x_2 , снова возможна соответствующая перестановка и исключение неизвестного x_2 , начиная с третьего уравнения, и т. д., вплоть до последнего шага прямого хода в методе Гаусса.

Заметим, что процедура прямого хода в методе Гаусса может привести не к треугольной матрице типа (6), а к двум другим случаям:

1) число преобразованных уравнений системы меньше числа неизвестных (это происходит, если в процессе преобразований получаются тождества $0 = 0$) — тогда система имеет бесконечное множество решений;

2) все коэффициенты при неизвестных в каком-либо уравнении равны нулю, в то время как свободный член уравнения отличен от нуля — тогда система не имеет решения.

Упражнения

Методом Гаусса решить системы линейных алгебраических уравнений $Ax = b$. Сравнить с точным решением ξ .

$$1. \quad A = \begin{bmatrix} 5 & 0 & 1 \\ 1 & 3 & -1 \\ -3 & 2 & 10 \end{bmatrix}, \quad b = \begin{bmatrix} 11 \\ 4 \\ 6 \end{bmatrix}, \quad \xi = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}.$$

$$2. \quad A = \begin{bmatrix} 2 & 0 & -1 \\ -1 & 3 & 1 \\ 1 & -1 & 4 \end{bmatrix}, \quad b = \begin{bmatrix} -3 \\ 2 \\ 3 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}.$$

$$3. \quad A = \begin{bmatrix} 2 & 0 & -1 \\ 1 & -3 & 1 \\ 1 & 1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}.$$

$$4. \quad A = \begin{bmatrix} 5 & 1 & -1 \\ -1 & 3 & 1 \\ 1 & -2 & 4 \end{bmatrix}, \quad b = \begin{bmatrix} -5 \\ 5 \\ 1 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}.$$

$$5. \quad A = \begin{bmatrix} 3 & 1 & -1 \\ -2 & 4 & 1 \\ 1 & 1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} -1 \\ 5 \\ -3 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}.$$

$$6. \quad A = \begin{bmatrix} 3 & 1 & -1 \\ 2 & 4 & 1 \\ 1 & -1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 6 \\ 9 \\ 4 \end{bmatrix}, \quad \xi = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}.$$

$$7. \quad A = \begin{bmatrix} 2 & -1 & 0 \\ 2 & 5 & -2 \\ 1 & -1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} -2 \\ -4 \\ 2 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}.$$

$$8. \quad A = \begin{bmatrix} 3 & -1 & 1 \\ 0 & 2 & -1 \\ -1 & 1 & 5 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 3 \\ -5 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}.$$

$$9. \quad A = \begin{bmatrix} 4 & 1 & -1 \\ 2 & 3 & 0 \\ 1 & -1 & 5 \end{bmatrix}, \quad b = \begin{bmatrix} 7 \\ 7 \\ 11 \end{bmatrix}, \quad \xi = \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix}.$$

$$10. \quad A = \begin{bmatrix} 2 & 0 & -1 \\ 1 & -4 & 2 \\ 1 & 1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -5 \\ 6 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}.$$

$$11. \quad A = \begin{bmatrix} 2 & -1 & 0 \\ 0 & 5 & 2 \\ 1 & -1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 7 \\ 4 \end{bmatrix}, \quad \xi = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}.$$

$$12. \quad A = \begin{bmatrix} 3 & 1 & -1 \\ 1 & 5 & -1 \\ 2 & 0 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} -2 \\ 8 \\ 1 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix}.$$

$$13. \quad A = \begin{bmatrix} 3 & 0 & -1 \\ 2 & -5 & 1 \\ 2 & -2 & 6 \end{bmatrix}, \quad b = \begin{bmatrix} -4 \\ 9 \\ 8 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ -2 \\ 1 \end{bmatrix}.$$

$$14. \quad A = \begin{bmatrix} 3 & 0 & -1 \\ 2 & -5 & 1 \\ 2 & 2 & 5 \end{bmatrix}, \quad b = \begin{bmatrix} 7 \\ -2 \\ 1 \end{bmatrix}, \quad \xi = \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}.$$

$$15. \quad A = \begin{bmatrix} 3 & -1 & 1 \\ 3 & 5 & 1 \\ -1 & 2 & 4 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 12 \\ -1 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}.$$

$$16. \quad A = \begin{bmatrix} -4 & 2 & 1 \\ -1 & 5 & 1 \\ 2 & 0 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} -5 \\ -5 \\ 5 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}.$$

$$17. \quad A = \begin{bmatrix} 5 & -1 & 1 \\ -1 & 3 & 1 \\ 2 & 1 & 4 \end{bmatrix}, \quad b = \begin{bmatrix} -3 \\ -1 \\ 1 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}.$$

$$18. \quad A = \begin{bmatrix} 4 & 2 & 1 \\ 3 & 5 & -1 \\ 2 & 1 & -4 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 4 \\ 6 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}.$$

$$19. \quad A = \begin{bmatrix} 4 & 1 & 2 \\ 0 & 3 & 1 \\ 1 & 2 & 4 \end{bmatrix}, \quad b = \begin{bmatrix} -6 \\ -1 \\ -5 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 0 \\ -1 \end{bmatrix}.$$

$$20. \quad A = \begin{bmatrix} 3 & 1 & 1 \\ 0 & -2 & 1 \\ 2 & -1 & 4 \end{bmatrix}, \quad b = \begin{bmatrix} 6 \\ -3 \\ -1 \end{bmatrix}, \quad \xi = \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}.$$

$$21. \quad A = \begin{bmatrix} 2 & -1 & 0 \\ 2 & 5 & 1 \\ 2 & 1 & -4 \end{bmatrix}, \quad b = \begin{bmatrix} -5 \\ 2 \\ -7 \end{bmatrix}, \quad \xi = \begin{bmatrix} -2 \\ 1 \\ 1 \end{bmatrix}.$$

$$22. \quad A = \begin{bmatrix} 5 & 0 & 1 \\ 1 & -3 & 1 \\ 3 & -1 & 5 \end{bmatrix}, \quad b = \begin{bmatrix} 11 \\ 3 \\ 11 \end{bmatrix}, \quad \xi = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}.$$

$$23. \quad A = \begin{bmatrix} 5 & -1 & 3 \\ 1 & -3 & 1 \\ 0 & 1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ -6 \\ 0 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}.$$

$$24. \quad A = \begin{bmatrix} 5 & -1 & 1 \\ -2 & 5 & 1 \\ 3 & -1 & 5 \end{bmatrix}, \quad b = \begin{bmatrix} -6 \\ 13 \\ 0 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix}.$$

$$25. \quad A = \begin{bmatrix} 5 & 1 & -1 \\ -1 & 3 & 1 \\ -1 & 0 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 8 \\ 0 \\ -5 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}.$$

$$26. \quad A = \begin{bmatrix} 3 & -1 & 1 \\ -1 & 3 & 1 \\ 1 & -1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} -4 \\ 4 \\ 4 \end{bmatrix}, \quad \xi = \begin{bmatrix} -2 \\ 0 \\ 2 \end{bmatrix}.$$

$$27. \quad A = \begin{bmatrix} 1 & 3 & 2 & -1 \\ 5 & 1 & -1 & 0 \\ 2 & 5 & 1 & 1 \\ 0 & 1 & -1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} -3 \\ 4 \\ -2 \\ 1 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 1 \end{bmatrix}.$$

$$28. \quad A = \begin{bmatrix} 3 & 0 & -1 & 1 \\ 2 & 2 & 1 & -1 \\ 2 & -5 & 1 & 0 \\ 1 & 1 & -1 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 4 \\ -1 \\ 6 \\ 1 \end{bmatrix}, \quad \xi = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 0 \end{bmatrix}.$$

$$29. \quad A = \begin{bmatrix} 2 & 3 & -4 & 1 \\ 1 & -2 & -5 & 1 \\ 5 & -3 & 1 & -4 \\ 10 & 2 & -1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ -1 \\ -16 \\ -4 \end{bmatrix}, \quad \xi = \begin{bmatrix} -1 \\ 1 \\ 0 \\ 2 \end{bmatrix}.$$

где $\beta_i = \frac{b_i}{a_{ii}}$ ($i=1, 2, \dots, n$), $a_{ij} = \begin{cases} 0, & i=j; \\ -\frac{a_{ij}}{a_{ii}}, & i \neq j. \end{cases}$

Введем обозначения

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}, \quad \alpha = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \dots \\ \beta_n \end{bmatrix}$$

и перепишем систему (9) в виде одного матричного уравнения

$$\mathbf{x} = \alpha \mathbf{x} + \beta. \quad (10)$$

Здесь $\alpha \mathbf{x}$ — произведение матрицы α на вектор $\mathbf{x}$.

Последовательные приближения (итерации) найдем следующим образом. Возьмем в качестве начального приближения $\mathbf{x}^{(0)}$ вектор β и подставим его в правую часть уравнения (10); получим $\mathbf{x}^{(1)}$. Продолжая аналогичные вычисления, придем к векторной последовательности приближений:

$$\begin{aligned} \mathbf{x}^{(1)} &= \alpha \mathbf{x}^{(0)} + \beta && \text{— первое приближение,} \\ \mathbf{x}^{(2)} &= \alpha \mathbf{x}^{(1)} + \beta && \text{— второе приближение,} \\ &\dots && \dots \\ \mathbf{x}^{(k+1)} &= \alpha \mathbf{x}^{(k)} + \beta && \text{— } (k+1)\text{-е приближение,} \\ &\dots && \dots \end{aligned} \quad (11)$$

Если существует предел ξ последовательности векторов $\mathbf{x}^{(k)}$, то, переходя к пределу в равенстве $\mathbf{x}^{(k+1)} = \alpha \mathbf{x}^{(k)} + \beta$ при $k \rightarrow \infty$, убеждаемся, что ξ является решением уравнения (10), т.е.

$$\xi = \alpha \xi + \beta.$$

Достаточные условия сходимости итераций к решению содержит следующая теорема.

Теорема. Если какая-либо норма матрицы меньше единицы: $\|\alpha\| < 1$, то уравнение (10) имеет единственное решение ξ , к которому стремится последовательность итераций (11) при любом выборе начального приближения $\mathbf{x}^{(0)}$.

В расчетах полагают $\mathbf{x}^{(0)} = \beta$. Погрешность приближенного решения уравнения (10) на k -м шаге оценивают неравенством

$$\|\xi - \mathbf{x}^{(k)}\| \leq \frac{\|\alpha\|^{k+1}}{1 - \|\alpha\|} \|\beta\|. \quad (12)$$

Из неравенства (12) можно получить оценку числа итераций k , необходимых для обеспечения заданной точности ε .

Отклонение приближения $\mathbf{x}^{(k)}$ от решения ξ по норме не будет превышать ε , если

$$\|\xi - \mathbf{x}^{(k)}\| \leq \frac{\|\alpha\|}{1 - \|\alpha\|} \|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| \leq \frac{\|\alpha\|^{k+1}}{1 - \|\alpha\|} \|\beta\| \leq \varepsilon. \quad (13)$$

Неравенство (13) дает обычно завышенную оценку числа итераций k . В оценках (12), (13) одновременно используются согласованные нормы для матриц и векторов (m - и l -нормы).

Из неравенств (13) видно, что условие, позволяющее принять приближение $\mathbf{x}^{(k)}$ в качестве решения с точностью ε , можно представлять в следующей удобной для вычислительного процесса форме:

$$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| \leq \frac{1 - \|\alpha\|}{\|\alpha\|} \varepsilon. \quad (14)$$

Пример 1. Найти решение системы уравнений

$$\begin{cases} 5x_1 - x_2 + 2x_3 = 8, \\ x_1 + 4x_2 - x_3 = -4, \\ x_1 + x_2 + 4x_3 = 4 \end{cases}$$

методом итераций с точностью 10^{-2} .

Решение. Приведем данную систему к виду (9):

$$\begin{cases} x_1 = 0 \cdot x_1 + 0,2x_2 - 0,4x_3 + 1,6, \\ x_2 = -0,25x_1 + 0 \cdot x_2 + 0,25x_3 - 1, \\ x_3 = -0,25x_1 - 0,25x_2 + 0 \cdot x_3 + 1. \end{cases}$$

Теперь запишем последовательность итераций ($k = 0, 1, 2, \dots$):

$$\begin{cases} x_1^{(k+1)} = 0 \cdot x_1^{(k)} + 0,2x_2^{(k)} - 0,4x_3^{(k)} + 1,6, \\ x_2^{(k+1)} = -0,25x_1^{(k)} + 0 \cdot x_2^{(k)} + 0,25x_3^{(k)} - 1, \\ x_3^{(k+1)} = -0,25x_1^{(k)} - 0,25x_2^{(k)} + 0 \cdot x_3^{(k)} + 1. \end{cases} \quad (15)$$

Для приведенной матрицы

$$\alpha = \begin{bmatrix} 0 & 0,2 & -0,4 \\ -0,25 & 0 & 0,25 \\ -0,25 & -0,25 & 0 \end{bmatrix}$$

достаточное условие сходимости процесса итераций выполняется по m -норме, поскольку

$$\|\alpha\|_m = \max_{1 \leq i \leq 3} \left\{ \sum_{j=1}^3 |\alpha_{ij}| \right\} = \max \{0,6; 0,5; 0,5\} = 0,6 < 1.$$

В качестве начального приближения возьмем вектор-столбец свободных членов приведенной системы:

$$\mathbf{x}^{(0)} = \begin{bmatrix} 1,6 \\ -1 \\ 1 \end{bmatrix}.$$

Число итераций для достижения заданной точности $\varepsilon = 10^{-2}$ определим

из неравенства $\frac{\|\alpha\|^{k+1}}{1 - \|\alpha\|} \|\beta\| \leq \varepsilon$, которое перепишем так:

$$k \geq \frac{-2 + \lg 0,25}{\lg 0,6} - 1 \approx 11.$$

Здесь учтено, что $\|\alpha\|_m = 0,6$; $\|\beta\|_m = \|\mathbf{x}^{(0)}\|_m = 1,6$.

Вычислим теперь три последовательных приближения по формулам (15) и оценим погрешность каждого результата, используя неравенство (13) в виде

$$\|\xi - \mathbf{x}^{(k)}\|_m \leq \frac{\|\alpha\|_m}{1 - \|\alpha\|_m} \|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|_m.$$

Найдем первое приближение:

$$\begin{cases} x_1^{(1)} = 0 \cdot x_1^{(0)} + 0,2x_2^{(0)} - 0,4x_3^{(0)} + 1,6 = 0,2(-1) - 0,4 + 1,6 = 1, \\ x_2^{(1)} = -0,25x_1^{(0)} + 0 \cdot x_2^{(0)} + 0,25x_3^{(0)} - 1 = -0,25 \cdot 1,6 + 0,25 \cdot (-1) = -1,15, \\ x_3^{(1)} = -0,25x_1^{(0)} - 0,25x_2^{(0)} + 0 \cdot x_3^{(0)} + 1 = -0,25 \cdot 1,6 - 0,25(-1) + 1 = 0,85; \end{cases}$$

$$\|x^{(1)} - x^{(0)}\|_m = \max \{0,6; 0,15; 0,15\} = 0,6.$$

Следовательно, $x^{(1)} = \begin{bmatrix} 1 \\ -1,15 \\ 0,85 \end{bmatrix}$ дает значение корня ξ с погреш-

ностью, не превышающей величины $\frac{\|a\|_m}{1 - \|a\|_m} \|x^{(1)} - x^{(0)}\|_m = \frac{0,6}{1 - 0,6} 0,6 = 0,9$.

Далее последовательно находим

$$\begin{cases} x_1^{(2)} = 0,2(-1,15) - 0,4 \cdot 0,85 + 1,6 = 1,03, \\ x_2^{(2)} = -0,25 \cdot 1 + 0,25 \cdot 0,85 - 1 = -1,0375, \\ x_3^{(2)} = -0,25 \cdot 1 - 0,25(-1,15) + 1 = 1,0375; \end{cases}$$

$$\|x^{(2)} - x^{(1)}\|_m = \max \{0,03; 0,1125; 0,1875\} = 0,1875,$$

$$x^{(2)} = \begin{bmatrix} 1,03 \\ -1,0375 \\ 1,0375 \end{bmatrix}, \quad \frac{\|a\|_m}{1 - \|a\|_m} \|x^{(2)} - x^{(1)}\|_m = \frac{0,6}{1 - 0,6} 0,1875 = 0,28125;$$

$$\begin{cases} x_1^{(3)} = 0,2(-1,0375) - 0,4 \cdot 1,0375 + 1,6 = 0,9775, \\ x_2^{(3)} = -0,25 \cdot 1,03 + 0,25 \cdot 1,0375 - 1 = -0,998125, \\ x_3^{(3)} = -0,25 \cdot 1,03 - 0,25(-1,0375) + 1 = 1,001875; \end{cases}$$

$$\|x^{(3)} - x^{(2)}\|_m = \max \{0,0525; 0,039375; 0,035625\} = 0,0525 \approx 0,05,$$

$$x^{(3)} = \begin{bmatrix} 0,9775 \\ -0,998125 \\ 1,001875 \end{bmatrix}, \quad \frac{\|a\|_m}{1 - \|a\|_m} \|x^{(3)} - x^{(2)}\|_m = \frac{0,6}{1 - 0,6} 0,05 = 0,075.$$

Заметим, что точное решение есть $x_1=1$, $x_2=-1$, $x_3=1$.

Решение этого примера по программе, приведенной в Приложении (см. с. 204-219), показывает, что заданная точность достигается за 5 шагов. ■

Пример 2. Показать, что для системы

$$\begin{cases} 100x_1 + 30x_2 - 70x_3 = 1, \\ 15x_1 - 50x_2 - 5x_3 = 1, \\ 6x_1 + 2x_2 + 20x_3 = 1 \end{cases}$$

процесс итераций сходится. Сколько итераций следует выполнить, чтобы найти решение системы с точностью 10^{-3} ?

Решение. Представим систему в приведенном для итераций виде:

$$\begin{cases} x_1 = 0 \cdot x_1 - 0,3x_2 + 0,7x_3 + 0,01, \\ x_2 = 0,3x_1 + 0 \cdot x_2 - 0,1x_3 - 0,02, \\ x_3 = -0,3x_1 - 0,1x_2 + 0 \cdot x_3 + 0,05. \end{cases}$$

Матрица приведенной системы

$$\alpha = \begin{pmatrix} 0 & -0,3 & 0,7 \\ 0,3 & 0 & -0,1 \\ -0,3 & -0,1 & 0 \end{pmatrix}.$$

Для проверки достаточного условия сходимости вычислим нормы матрицы α :

$$\|\alpha\|_m = \max_{1 \leq i \leq 3} \left(\sum_{j=1}^3 |\alpha_{ij}| \right) = \max\{1; 0,4; 0,4\} = 1,$$

$$\|\alpha\|_l = \max_{1 \leq j \leq 3} \left(\sum_{i=1}^3 |\alpha_{ij}| \right) = \max\{0,6; 0,4; 0,8\} = 0,8,$$

$$\|\alpha\|_k = \sqrt{\sum_{i=1}^3 \sum_{j=1}^3 \alpha_{ij}^2} = \sqrt{0,3^2 + 0,7^2 + 0,3^2 + 0,1^2 + 0,3^2 + 0,1^2} = \sqrt{0,78} \approx 0,88.$$

Достаточное условие сходимости процесса итераций выполнено, так как, в частности, $\|\alpha\|_l = 0,8 < 1$. Чтобы найти решение системы с точностью $\varepsilon = 10^{-3}$, будем вычислять число итераций с помощью неравенства (13) в пространстве с l -нормой. Тогда получим

$$\|\beta\|_l = \sum_{i=1}^3 |\beta_i| = 0,01 + 0,02 + 0,05 = 0,08;$$

$$\frac{\|a\|_1^{k+1}}{1 - \|a\|_1} \leq \varepsilon \Rightarrow \frac{0,8^{k+1}}{0,2} 0,08 \leq 10^{-3} \Rightarrow 0,8^{k+1} \cdot 0,4 \leq 10^{-3} \Rightarrow$$

$$\Rightarrow (k+1) \lg 0,8 + \lg 0,4 \leq -3 \Rightarrow k \geq \frac{-3 - \lg 0,4}{\lg 0,8} - 1 \approx 25.$$

Фактическое число итераций для достижения заданной точности в пространстве с m -нормой равно 10 (расчет по программе, приведенной в Приложении). ■

Замечание. Достаточное условие сходимости процесса итераций для неприведенной системы (8) по m -норме можно представить в виде

$$|a_{ii}| > \sum_{\substack{j=1 \\ i \neq j}}^n |a_{ij}| \quad (i=1, 2, \dots, n). \quad (16)$$

Здесь модули диагональных коэффициентов для каждого уравнения системы (8) больше суммы модулей всех остальных коэффициентов. Естественно, что если условия (16) не выполнены, то следует применять условия сходимости по l - и k -нормам после приведения системы к виду (9) или непосредственно к неприведенной системе (8). Достаточное условие сходимости процесса итераций для неприведенной системы (8) по l -норме можно представить в виде

$$|a_{jj}| > \sum_{\substack{i=1 \\ i \neq j}}^n |a_{ij}| \quad (j=1, 2, \dots, n).$$

Упражнения

Найти решения систем линейных алгебраических уравнений $Ax=b$ методом итераций с точностью $\varepsilon=10^{-2}$ (см. упражнения к §1). Сравнить их с точными решениями §. Для систем 1-26 выполнить вручную первые две итерации. Убедиться, что для систем 27-34 достаточные условия сходимости метода итераций не выполнены.

МЕТОДЫ ЧИСЛЕННОГО РЕШЕНИЯ УРАВНЕНИЙ И СИСТЕМ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

§1. ОТДЕЛЕНИЕ КОРНЕЙ

Пусть $f(x) = 0$ — некоторое уравнение. Число ξ называется **корнем** или **решением** данного уравнения, если оно, будучи подставлено в уравнение, обращает его в равенство, т.е. $f(\xi)=0$. Число ξ называют **нулем функции** $y=f(x)$.

Нахождение действительных корней с определенной точностью можно разбить на два этапа:

- 1) отделение корней, т.е. установление промежутков, в которых содержится один корень уравнения;
- 2) вычисление корня, принадлежащего выбранному промежутку, с заданной точностью.

Известно, что если функция $f(x)$ непрерывна и принимает на концах отрезка $[a, b]$ значения разных знаков, т.е. $f(a)f(b)<0$, то внутри этого промежутка найдется нуль функции.

Отделение корней уравнения $f(x)=0$ для непрерывной в области определения функции $f(x)$ можно осуществить различными способами.

- 1) Составляют таблицу значений функции $y=f(x)$ на определенном промежутке изменения аргумента x , и если окажется, что для соседних значений аргументов значения функции имеют разные знаки, то нуль функции находится между ними.

- 2) Уравнение $f(x)=0$ заменяют равносильным $\varphi(x) = \psi(x)$. Строят графики функций $y=\varphi(x)$ и $y=\psi(x)$; искомый корень является абсциссой точки пересечения этих графиков.

- 3) Строят график функции $y=f(x)$ на промежутке изменения x ; тогда абсцисса ξ точки пересечения графика с осью Ox — нуль функции, т.е. $f(\xi) = 0$.

Пример. Выяснить, сколько корней имеет уравнение $4-e^x-2x^2=0$, и найти промежутки, в которых находятся эти корни.

Решение. Рассмотрим три функции:

$$f(x) = 4 - e^x - 2x^2; \quad \varphi(x) = 4 - 2x^2; \quad \psi(x) = e^x.$$

Уравнение $f(x) = 0$ эквивалентно уравнению $\varphi(x) = \psi(x)$. Отделим его корни двумя способами.

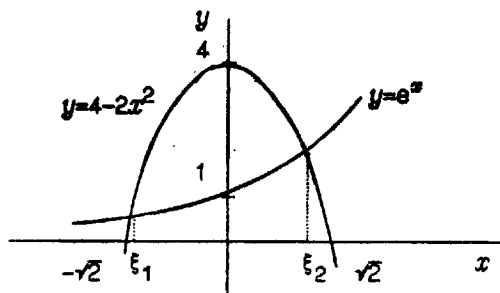


Рис. 10

1. Из таблицы значений функции $f(x)$ на промежутке $[-3, 0; 1, 0]$ с шагом изменения x , равным 1, видно, что существуют корни уравнения на отрезках

x	$f(x)$	$\varphi(x)$	$\psi(x)$
-3,0	-14,05	-14,00	0,05
-2,0	-4,14	-4,00	0,14
-1,0	1,63	2,00	0,37
0,0	3,00	4,00	1,00
1,0	-0,72	2,00	2,72

$[-2, -1]$ и $[0, 1]$, так как значения функции на концах отрезка имеют разные знаки.

2. Графики функций $y=\varphi(x)$ и $y=\psi(x)$ пересекаются в двух точках, абсциссы которых ξ_1 и ξ_2 являются решениями уравнения $\varphi(x)=\psi(x)$, заключенными в указанных промежутках (рис.10). ■

§2. МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ ДЛЯ УРАВНЕНИЯ $f(x)=0$

Пусть дано уравнение

$$f(x) = 0, \quad (1)$$

причем функция $f(x)$ непрерывна на отрезке $[a, b]$ и $f(a)f(b) < 0$ (рис.11). Для вычисления корня уравнения (1), принадлежащего

отрезку $[a, b]$, найдем середину этого отрезка $x_1 = \frac{a+b}{2}$. Если

$f(x_1) \neq 0$, то для продолжения вычислений выберем ту из частей данного отрезка $[a, x_1]$ или $[x_1, b]$, на концах которой функция $f(x)$ имеет противоположные знаки. Концы нового отрезка обозначим через a_1 и b_1 (рис.11).

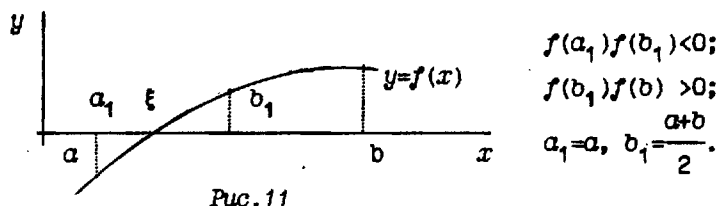


Рис. 11

Новый суженный промежуток $[a_1, b_1]$ снова делим пополам и проводим вычисления по разобранной схеме и т. д. В результате получаем либо точный корень уравнения (1) на каком-то этапе, либо последовательность вложенных отрезков $[a, b]$, $[a_1, b_1]$, ..., $[a_n, b_n]$, ... таких, что

$$f(a_n)f(b_n) < 0 \quad (n = 1, 2, \dots), \quad (2)$$

$$b_n - a_n = \frac{1}{2^n} (b-a). \quad (3)$$

Число ξ — общий предел последовательностей $\{a_n\}$ и $\{b_n\}$ — является корнем уравнения $f(x)=0$.

Оценку погрешности решения на n -м шаге вычислений можно получить из соотношения (3) в виде

$$0 \leq \xi - a_n \leq \frac{1}{2^n} (b-a) = b_n - a_n. \quad (4)$$

Здесь $a_n \approx \xi$ с точностью ϵ , не превышающей $\frac{1}{2^n} (b-a)$.

Пример. Методом половинного деления с точностью $\epsilon = 10^{-2}$ найти корень уравнения $4 - e^x - 2x^2 = 0 \quad (x > 0)$. $\{ \}$

Решение. В примере из §1 при отделении корней уравнения было установлено, что искомый корень ξ принадлежит отрезку $[0, 1]$. На каждом шаге вычислений значение корня принимаем равным $x_n = (a_n + b_n)/2$ с погрешностью $d_n = b_n - a_n$. Будем производить вычисления и выбирать последовательность вложенных отрезков $[a_n, b_n]$, используя условие $f(a_n)f(b_n) < 0$. Имеем

$$[a, b] = [0, 1], \quad x_1 = \frac{a+b}{2} = 0,5.$$

Так как $f(a) = 3$, $f(x_1) = 1,8513$ и $f(a)f(x_1) > 0$, то полагаем $a_1 = x_1 = 0,5$, $b_1 = b = 1$; $d_1 = b_1 - a_1 = 0,5$. Тогда

$$[a_1, b_1] = [0, 5; 1], \quad x_2 = \frac{a_1 + b_1}{2} = \frac{0,5 + 1}{2} = 0,75.$$

Здесь $f(a_1) = 1,8513$, $f(x_2) = 0,758$, $f(a_1)f(x_2) > 0$; следовательно, $a_2 = x_2 = 0,75$, $b_2 = b_1 = 1$; $d_2 = b_2 - a_2 = 0,25$. Тогда

$$[a_2, b_2] = [0,75; 1], \quad x_3 = \frac{a_2 + b_2}{2} = 0,875; \quad d_3 = 0,125.$$

Производя вычисления далее или воспользовавшись программой Приложения (см. с. 220–225), можно убедиться, что заданная точность достигается на 7-м шаге:

$$x_7 = 0,8828125 \text{ с погрешностью } d_7 = 0,0078125 < \epsilon = 0,01. \blacksquare$$

Упражнения

Методом половинного деления найти решения следующих уравнений с точностью $\epsilon = 10^{-2}$.

1. $x^4 - 3x - 20 = 0 \quad (x > 0)$.
2. $x^3 - 2x - 5 = 0 \quad (x > 0)$.
3. $x^3 + 3x + 5 = 0$.
4. $x^4 + 5x - 7 = 0 \quad (x > 0)$.
5. $x^3 - 12x - 5 = 0 \quad (x > 0)$.
6. $x^3 - 2x^2 - 4x + 5 = 0 \quad (x < 0)$.
7. $x + e^x = 0$.
8. $x^5 - x - 2 = 0$.
9. $x^3 - 10x + 5 = 0 \quad (x < 0)$.
10. $2 - \ln x - x = 0$.
11. $x^3 + 2x - 7 = 0$.
12. $x^3 + x^2 - 11 = 0 \quad (x > 0)$.
13. $x^4 - 2x - 4 = 0 \quad (x > 0)$.
14. $2e^x + x - 1 = 0$.
15. $x^4 - 2x - 4 = 0 \quad (x < 0)$.
16. $2x^3 + x^2 - 4 = 0 \quad (x > 0)$.
17. $e^x - x - 2 = 0$.
18. $\frac{1}{2}e^x - x - 1 = 0 \quad (x > 0)$.
19. $x^2 - \cos x = 0 \quad (x > 0)$.
20. $x^2 + \ln x = 0$.
21. $\ln x + 0,5x - 1 = 0$.
22. $\ln x - 0,5x + 1 = 0 \quad (x > 1)$.
23. $\frac{1}{1+x^2} - \ln x = 0$.
24. $\frac{1}{1+x^2} - \frac{e^x}{2} = 0 \quad (x > 0)$.
25. $\frac{x}{2+x} - \ln x = 0$.

§3. МЕТОД ИТЕРАЦИЙ ДЛЯ ОДНОГО УРАВНЕНИЯ

С ОДНИМ НЕИЗВЕСТНЫМ

Пусть требуется решить уравнение, представленное в виде

$$x = g(x), \quad (5)$$

где правая часть уравнения — непрерывная на отрезке $[a, b]$ функция $g(x)$. Суть метода итераций (метода последовательных приближений) состоит в следующем. Начиная с произвольной точки $x^{(0)}$, принадлежащей отрезку $[a, b]$, последовательно получаем

$$x^{(1)} = g(x^{(0)}) \quad (\text{первое приближение}),$$

$$x^{(2)} = g(x^{(1)}) \quad (\text{второе приближение}),$$

$$\dots \dots \dots$$

$$x^{(k+1)} = g(x^{(k)}) \quad (k+1 \text{ — } \text{приближение}),$$

$$\dots \dots \dots$$

Последовательность

$$x^{(0)}, x^{(1)}, \dots, x^{(k)}, \dots \quad (6)$$

называется **последовательностью итераций** для уравнения (5) с начальной точкой $x^{(0)}$. Если все точки (6) принадлежат отрезку $[a, b]$ и существует предел $\xi = \lim_{k \rightarrow \infty} x^{(k)}$, то, перейдя к пределу в равенстве

$$x^{(k+1)} = g(x^{(k)}) \quad (k = 0, 1, 2, \dots), \quad (7)$$

получим $\lim_{k \rightarrow \infty} x^{(k+1)} = \lim_{k \rightarrow \infty} g(x^{(k)})$, т.е. $\xi = g(\xi)$.

Следовательно, если существует предел последовательности итераций (6), то он является корнем уравнения (5). Достаточные условия сходимости последовательности итераций содержатся в следующей теореме.

Теорема. Пусть функция $g(x)$ имеет на отрезке $[a, b]$ непрерывную производную и выполнены два условия:

- 1) $|g'(x)| \leq q < 1$ при $x \in [a, b]$;
- 2) значения функции $y = g(x)$ принадлежат отрезку $[a, b]$ для любого $x \in [a, b]$.

Тогда при любом выборе начального приближения $x^{(0)} \in [a, b]$ процесс итераций сходится к единственному корню ξ уравнения (5) на отрезке $[a, b]$.

Оценка погрешности k -го приближения $x^{(k)}$ к корню ξ такова:

$$|\xi - x^{(k)}| \leq \frac{q}{1-q} |x^{(k)} - x^{(k-1)}|, \quad (8)$$

где
$$q = \max_{a \leq x \leq b} |g'(x)|.$$

Укажем теперь один из способов преобразования уравнения

$$f(x) = 0 \quad (9)$$

к виду $x = g(x)$, допускающему применение метода итераций, сходящихся к решению ξ уравнения (9).

Для любого числа $\lambda \neq 0$ уравнение (9) равносильно уравнению (5), где $g(x) = x + \lambda f(x)$. Предположим, что производная $f'(x) > 0$ и непрерывна на $[a, b]$. Пусть $M = \max_{a \leq x \leq b} f'(x)$, $m = \min_{a \leq x \leq b} f'(x)$;

положим $\lambda = -\frac{1}{M}$, $q = 1 - \frac{m}{M}$ и рассмотрим функцию

$$g(x) = x - \frac{1}{M} f(x). \quad (10)$$

Для функции, определенной формулой (10), выполняются достаточные условия сходимости метода итераций решения уравнения (9). В частности, условие 1 теоремы следует из неравенств

$$0 < m \leq f'(x) \leq M,$$

$$0 \leq g'(x) = 1 - \frac{1}{M} f'(x) \leq 1 - \frac{m}{M} = q < 1 \quad \forall x \in [a, b].$$

Замечание 1. Если окажется, что производная $f'(x)$ отрицательна на отрезке $[a, b]$, то уравнение (5) можно заменить на равносильное уравнение $-f(x) = 0$ и использовать указанное преобразование.

Замечание 2. Если вычисление точного значения числа $M = \max_{a \leq x \leq b} |f'(x)|$ затруднительно, то можно заменить его произвольным числом $M_1 > M$. Однако при большом M_1 число $q = 1 - \frac{m}{M_1}$ ближе к единице и процесс итераций сходится медленнее.

Замечание 3. При нахождении корня уравнения (5) с заданной точностью $\epsilon > 0$ или при оценке погрешности k -го приближения можно, не вычисляя точного значения числа $q = \max_{a \leq x \leq b} |g'(x)|$, ограничиться следующей практической рекомендацией:

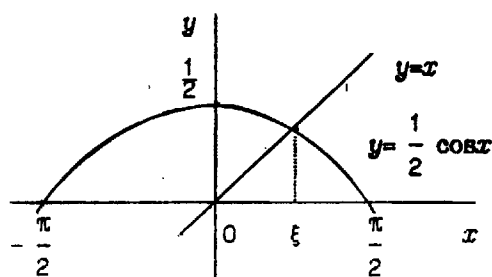


Рис. 12

$$|\xi - x^{(k)}| \leq \begin{cases} |x^{(k)} - x^{(k-1)}| \leq \varepsilon & \text{при } 0 < q \leq \frac{1}{2}; \\ 10|x^{(k)} - x^{(k-1)}| \leq \varepsilon & \text{при } \frac{1}{2} < q < 1. \end{cases} \quad (11a)$$

$$(11b)$$

Пример 1. Решить с точностью $\varepsilon = 10^{-2}$ уравнение

$$2x - \cos x = 0. \quad (12)$$

Решение. Для отделения корней представим уравнение (12) в виде

$$x = \frac{1}{2} \cos x.$$

Построив графики функций $y=x$ и $y = \frac{1}{2} \cos x$ (рис. 12), видим, что корень уравнения (12) содержится внутри отрезка $\left[0, \frac{\pi}{2}\right]$.

Здесь $f(x) = 2x - \cos x$; $f'(x) = 2 - \sin x > 0$; $M = \max_{0 \leq x \leq \frac{\pi}{2}} f'(x) = 2$,

$$\lambda = -\frac{1}{M} = -\frac{1}{2} \quad \text{и} \quad g(x) = x + \lambda f(x) = x - \frac{1}{2}(2x - \cos x) = \frac{1}{2} \cos x.$$

Положим $x^{(0)} = 0,5$. Последовательные приближения найдем по формулам

$$x^{(k+1)} = \frac{1}{2} \cos x^{(k)} \quad (k=0,1,2,\dots):$$

$$x^{(1)} = \frac{1}{2} \cos x^{(0)} = \frac{1}{2} \cos 0,5 = 0,43879128;$$

$$x^{(2)} = \frac{1}{2} \cos x^{(1)} = \frac{1}{2} \cos 0,43879128 = 0,45263292;$$

$$x^{(3)} = \frac{1}{2} \cos x^{(2)} = \frac{1}{2} \cos 0,45263292 = 0,44964938;$$

$$x^{(4)} = \frac{1}{2} \cos x^{(3)} = \frac{1}{2} \cos 0,44964938 = 0,45029978.$$

Для оценки погрешности четвертого приближения воспользуемся неравенством (11а). Так как $q = \max_{0 < x < \frac{\pi}{2}} |g'(x)| = \frac{1}{2} \max_{0 < x < \frac{\pi}{2}} \sin x = \frac{1}{2}$, то

$$|\xi - x^{(4)}| \leq |x^{(4)} - x^{(3)}| = 0,0006504 < \varepsilon = 10^{-3}.$$

Следовательно, $\xi \approx x^{(4)} \approx 0,450$ с точностью 10^{-3} . ■

Пример 2. Решить с точностью $\varepsilon = 10^{-3}$ уравнение

$$x + \ln x = 0. \quad (13)$$

Решение. Представим уравнение (13) в виде $-x = \ln x$. Построив графики функций $y = -x$ и $y = \ln x$ (рис.13), заключаем, что уравнение (13) имеет единственный корень на отрезке $[0,5; 1]$. Иначе говоря, для непрерывной и монотонно возрастающей функции $f(x) = x + \ln x$, имеющей на концах отрезка значения разных знаков: $f(0,5) = -0,193$, $f(1) = 1$, существует единственный корень уравнения (13) на этом отрезке (функция монотонно возрастает, так как производная $f'(x) = 1 + \frac{1}{x}$ положительна в области определения). Вычислим

$$m = \min_{0,5 \leq x \leq 1} f'(x) = \min_{0,5 \leq x \leq 1} \left[1 + \frac{1}{x} \right] = 2;$$

$$M = \max_{0,5 \leq x \leq 1} f'(x) = \max_{0,5 \leq x \leq 1} \left[1 + \frac{1}{x} \right] = 3; \quad q = 1 - \frac{m}{M} = \frac{1}{3}.$$

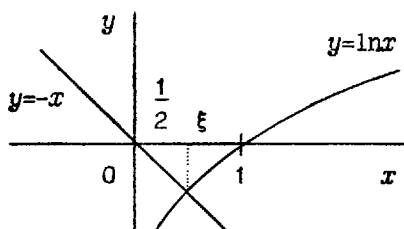


Рис. 13

Преобразуем исходное уравнение (13) к виду, удобному для итераций:

$$x = g(x); \quad g(x) = x - \frac{1}{M} f(x) = x - \frac{1}{3} (x + \ln x) = \frac{1}{3} (2x - \ln x).$$

Таким образом, сходящаяся к решению уравнения (13) последовательность итераций определяется из соотношений

$$x^{(k+1)} = \frac{1}{3} (2x^{(k)} - \ln x^{(k)}) \quad (k=0, 1, 2, \dots).$$

Пусть $x^{(0)} = 0,75$ — начальная точка в итерационном процессе. Оценку погрешности каждого приближения будем определять расстоянием $d_k = |x^{(k)} - x^{(k-1)}|$.

Производя вычисления, последовательно находим

$$x^{(1)} = \frac{1}{3} (2x^{(0)} - \ln x^{(0)}) = \frac{1}{3} (2 \cdot 0,75 - \ln 0,75) = 0,59589402, \\ d_1 = 0,1541106;$$

$$x^{(2)} = \frac{1}{3} (2 \cdot 0,59589402 - \ln 0,59589402) = 0,56982683, \\ d_2 = 0,0260672;$$

$$x^{(3)} = g(x^{(2)}) = 0,56735881, \quad d_3 = 0,00246805;$$

$$x^{(4)} = g(x^{(3)}) = 0,56716032.$$

Оценку погрешности четвертого приближения получим из неравенства

$$|\xi - x^{(4)}| \leq d_4 = |x^{(4)} - x^{(3)}| = 0,00019848 < \epsilon = 10^{-3}.$$

Следовательно, $\xi \approx x^{(4)} = 0,567$. ■

Пример 3. Решить методом итераций с точностью $\epsilon = 10^{-2}$ уравнение

$$4 - e^x - 2x^2 = 0 \quad (x > 0). \quad (14)$$

Решение. Уравнение (14) было решено в §2 методом половинного деления. Корень уравнения ξ принадлежит отрезку $[0, 1]$. Здесь производная $f'(x) = -e^x - 4x < 0$ при $x \in [0, 1]$ и является монотонной функцией, модуль которой достигает максимального и минимального значений на концах отрезка:

$$M = \max_{0 \leq x \leq 1} |f'(x)| = \max \{1; e+4\} = e+4 \approx 7;$$

$$\pi = \min_{0 < x < 1} |f'(x)| = 1; \quad q \approx 1 - \frac{1}{7} = \frac{6}{7}.$$

Так как в этом примере производная $f'(x)$ отрицательна на отрезке, то, преобразуя уравнение (14) к виду удобному для итераций, необходимо воспользоваться замечанием 1 (см. с. 41) следующим образом:

$$-f(x) = -4 + e^x + 2x^2; \quad g(x) = x - \frac{1}{M} (-f(x));$$

$$g(x) = x - \frac{1}{7} (-4 + e^x + 2x^2) = x + \frac{1}{7} (4 - e^x - 2x^2).$$

Последовательность вычислений определяется формулами

$$x^{(k+1)} = x^{(k)} + \frac{1}{7} \left[4 - e^{x^{(k)}} - 2(x^{(k)})^2 \right] \quad (k=0, 1, 2, \dots).$$

Для оценки погрешности k -го приближения здесь используем неравенство (116) в виде $|x^{(k)} - x^{(k-1)}| < \frac{\varepsilon}{10}$ ($k=1, 2, \dots$). Непос-

редственно можно убедиться, что если положить $x^{(0)} = 0,5$, то последнее неравенство будет выполняться, начиная с 5-й итерации.

Следовательно, $\xi \approx x^{(5)} = 0,887$ ($\varepsilon = 10^{-2}$). ■

Упражнения

Методом итераций с точностью $\varepsilon = 10^{-2}$ решить уравнения 1-25 (см. упражнения к §2).

§4. МЕТОД ИТЕРАЦИЙ ДЛЯ СИСТЕМЫ ДВУХ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

Систему двух уравнений с двумя неизвестными

$$\begin{cases} f_1(x_1, x_2) = 0, \\ f_2(x_1, x_2) = 0 \end{cases} \quad (15)$$

будем представлять в виде

$$\begin{cases} x_1 = g_1(x_1, x_2), \\ x_2 = g_2(x_1, x_2). \end{cases} \quad (16)$$

Используя векторные обозначения

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \mathbf{g}(\mathbf{x}) = \begin{bmatrix} g_1(x_1, x_2) \\ g_2(x_1, x_2) \end{bmatrix}, \quad (17)$$

перепишем систему (16) в компактной форме:

$$\mathbf{x} = \mathbf{g}(\mathbf{x}). \quad (18)$$

Решением системы уравнений (15) или (16) или (18) называют вектор $\xi = \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix}$, координаты которого, будучи подставлены в уравнения (15) или (16), обращают их в равенства.

Пусть предварительно установлено, что уравнение (18) имеет единственный корень ξ , принадлежащий замкнутому прямоугольнику

$$D = \{(x_1, x_2): a_1 \leq x_1 \leq b_1; a_2 \leq x_2 \leq b_2\}. \quad (19)$$

Возьмем произвольную точку $\mathbf{x}^{(0)} = \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \end{bmatrix}$, $\mathbf{x}^{(0)} \in D$ и, используя формулы

$$\mathbf{x}^{(k+1)} = \mathbf{g}(\mathbf{x}^{(k)}) \quad (k=0, 1, 2, \dots), \quad (20)$$

т.е.

$$\begin{cases} x_1^{(k+1)} = g_1(x_1^{(k)}, x_2^{(k)}), \\ x_2^{(k+1)} = g_2(x_1^{(k)}, x_2^{(k)}), \end{cases} \quad (21)$$

получим последовательность векторов

$$\mathbf{x}^{(k)} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \end{bmatrix} \quad (k=1, 2, \dots). \quad (22)$$

Последовательность (22) сходится к решению уравнения (18), если выполняются условия следующей теоремы.

Теорема. Пусть функции $g_1(x_1, x_2)$ и $g_2(x_1, x_2)$ — правые части уравнений (16) — непрерывны вместе со своими частными производными первого порядка в замкнутом прямоугольнике (19) и выполнены два условия:

- 1) норма матрицы Якоби функций $g_1(x_1, x_2)$ и $g_2(x_1, x_2)$ не превосходит единицы для любого вектора $\mathbf{x} \in D$;
- 2) значения вектор-функции $\mathbf{g}(\mathbf{x})$ принадлежат прямоугольнику D для любого вектора $\mathbf{x} \in D$.

Тогда при любом выборе начального приближения $\mathbf{x}^{(0)} \in D$ процесс итераций сходится к единственному корню ξ уравнения (18) в прямоугольнике D .

Оценка погрешности k -го приближения $\mathbf{x}^{(k)}$ к корню ξ такова:

$$\|\xi - \mathbf{x}^{(k)}\| \leq \frac{q}{1-q} \|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|, \quad (23)$$

где
$$q = \max_{\mathbf{x} \in D} \|J(\mathbf{x})\|.$$

Замечание 1. Для определенности в качестве нормы вектора $\mathbf{x}$ и соответствующей нормы матрицы Якоби $J(\mathbf{x})$ функций $g_1(x_1, x_2)$,

$$g_2(x_1, x_2), \text{ т.е. для } J(\mathbf{x}) = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} \end{bmatrix}, \text{ возьмем } m\text{-нормы:}$$

$$\|\mathbf{x}\| = \|\mathbf{x}\|_m = \max_{\mathbf{x} \in D} (|x_1|; |x_2|);$$

$$\|J(\mathbf{x})\| = \|J(\mathbf{x})\|_m = \max_{\mathbf{x} \in D} \left\{ \left| \frac{\partial g_1}{\partial x_1} \right| + \left| \frac{\partial g_1}{\partial x_2} \right|; \left| \frac{\partial g_2}{\partial x_1} \right| + \left| \frac{\partial g_2}{\partial x_2} \right| \right\}. \quad (24)$$

Замечание 2. При решении методом итераций системы нелинейных уравнений (15) областью D (в условиях теоремы) можно считать множество точек вблизи точки пересечения кривых, определяемых уравнениями $f_1(x_1, x_2) = 0$ и $f_2(x_1, x_2) = 0$.

Замечание 3. Для практической оценки погрешности k -го приближения можно пользоваться неравенствами, аналогичными (11a)-(11б):

$$\|\xi - \mathbf{x}^{(k)}\| \leq \begin{cases} \|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| \leq \varepsilon & \text{при } 0 < q \leq \frac{1}{2}; \\ 10\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| \leq \varepsilon & \text{при } \frac{1}{2} < q < 1. \end{cases} \quad (25a) \quad (25b)$$

Укажем способ преобразования системы (15) к системе (16), допускающий применение метода итераций. Преобразуем систему (15) так, чтобы в окрестности начальной точки $\mathbf{x}^{(0)}$, близкой к искомому решению, выполнялось условие 1 теоремы о достаточных условиях сходимости итераций к решению, т.е.

$$\|J(\mathbf{x})\| < 1. \quad (26)$$

Представим правые части уравнений (16) в виде

$$g_1(x_1, x_2) = x_1 + \lambda_{11}f_1(x_1, x_2) + \lambda_{12}f_2(x_1, x_2), \quad (27)$$

$$g_2(x_1, x_2) = x_2 + \lambda_{21}f_1(x_1, x_2) + \lambda_{22}f_2(x_1, x_2).$$

Здесь $f_1(x_1, x_2)$, $f_2(x_1, x_2)$ — правые части уравнений системы (15), по предположению, непрерывно дифференцируемые функции в прямоугольнике D . Система уравнений (15) равносильна системе (16) с правыми частями (27) при условии, что определитель, составленный из постоянных λ_{ij} ($i, j = 1, 2$), отличен от нуля, т.е.

$$\begin{vmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{vmatrix} = \lambda_{11} \lambda_{22} - \lambda_{12} \lambda_{21} \neq 0.$$

Для вычисления постоянных λ_{ij} предположим, что норма матрицы Якоби равна нулю в точке $\mathbf{x}^{(0)}$; тогда, в силу непрерывности компонент матрицы, найдется окрестность точки $\mathbf{x}^{(0)}$, где $\|J(\mathbf{x})\| < 1$, и условие 1 теоремы будет удовлетворено.

Равенство $\|J(\mathbf{x}^{(0)})\| = 0$ в соответствии с формулой (24) означает, что матрица $J(\mathbf{x}^{(0)})$ — нулевая, т.е.

$$\left. \frac{\partial g_1}{\partial x_1} \right|_{\mathbf{x}^{(0)}} = \left. \frac{\partial g_1}{\partial x_2} \right|_{\mathbf{x}^{(0)}} = \left. \frac{\partial g_2}{\partial x_1} \right|_{\mathbf{x}^{(0)}} = \left. \frac{\partial g_2}{\partial x_2} \right|_{\mathbf{x}^{(0)}} = 0. \quad (28)$$

Используя формулы (27), перепишем соотношения (28) в виде

$$\left\{ \begin{array}{l} \left\{ \begin{array}{l} 1 + \lambda_{11} \left. \frac{\partial f_1}{\partial x_1} \right|_{\mathbf{x}^{(0)}} + \lambda_{12} \left. \frac{\partial f_2}{\partial x_1} \right|_{\mathbf{x}^{(0)}} = 0, \\ \lambda_{11} \left. \frac{\partial f_1}{\partial x_2} \right|_{\mathbf{x}^{(0)}} + \lambda_{12} \left. \frac{\partial f_2}{\partial x_2} \right|_{\mathbf{x}^{(0)}} = 0; \end{array} \right. \\ \left\{ \begin{array}{l} \lambda_{21} \left. \frac{\partial f_1}{\partial x_1} \right|_{\mathbf{x}^{(0)}} + \lambda_{22} \left. \frac{\partial f_2}{\partial x_1} \right|_{\mathbf{x}^{(0)}} = 0, \\ 1 + \lambda_{21} \left. \frac{\partial f_1}{\partial x_2} \right|_{\mathbf{x}^{(0)}} + \lambda_{22} \left. \frac{\partial f_2}{\partial x_2} \right|_{\mathbf{x}^{(0)}} = 0. \end{array} \right. \end{array} \right. \quad (29)$$

В результате получим систему из четырех линейных уравнений отно-

сительно неизвестных λ_{ij} ($i, j = 1, 2$), которая эквивалентна двум независимо решаемым системам с одной и той же матрицей

$$\begin{bmatrix} \left. \frac{\partial f_1}{\partial x_1} \right|_{\mathbf{x}^{(0)}} & \left. \frac{\partial f_2}{\partial x_1} \right|_{\mathbf{x}^{(0)}} \\ \left. \frac{\partial f_1}{\partial x_2} \right|_{\mathbf{x}^{(0)}} & \left. \frac{\partial f_2}{\partial x_2} \right|_{\mathbf{x}^{(0)}} \end{bmatrix}$$

и разными правыми частями.

Пример. Дана система уравнений

$$\begin{cases} x_2(x_1 - 1) - 1 = 0, & (l_1) \\ x_1^2 - x_2^2 - 1 = 0. & (l_2) \end{cases} \quad (30)$$

Найти с точностью $\varepsilon = 10^{-3}$ ее решение, расположенное в первой четверти плоскости Ox_1x_2 .

Решение. Кривые, определяемые уравнениями (30), изображены на рис.14. Эти кривые пересекаются в двух точках ξ_1 и ξ_2 . Приведем систему (30) к виду, удобному для итераций (27), и найдем решение ξ_1 с заданной точностью.

Возьмем в качестве начального значения (графическое решение)

$\mathbf{x}^{(0)} = \begin{bmatrix} 1,5 \\ 1,5 \end{bmatrix}$ и для определения коэффициентов λ_{ij} будем решать

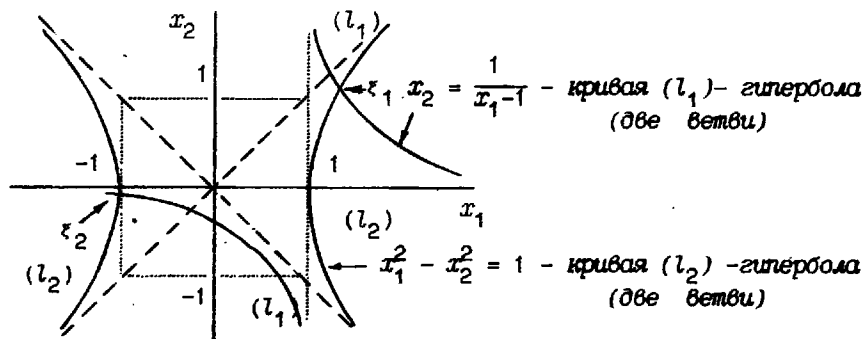


Рис.14

систему уравнений (29). Вычислим частные производные функций

$$f_1(x_1, x_2) = x_2(x_1 - 1) - 1, \quad f_2(x_1, x_2) = x_1^2 - x_2^2 - 1$$

в точке $x^{(0)}$:

$$\left. \frac{\partial f_1}{\partial x_1} \right|_{x^{(0)}} = x_2 \Big|_{x^{(0)}} = 1,5;$$

$$\left. \frac{\partial f_1}{\partial x_2} \right|_{x^{(0)}} = x_1 \Big|_{x^{(0)}} = 3;$$

$$\left. \frac{\partial f_1}{\partial x_2} \right|_{x^{(0)}} = (x_1 - 1) \Big|_{x^{(0)}} = 0,5;$$

$$\left. \frac{\partial f_2}{\partial x_2} \right|_{x^{(0)}} = -2x_2 \Big|_{x^{(0)}} = -3.$$

Решив систему

$$\begin{cases} \begin{cases} 1 + 1,5 \lambda_{11} + 3 \lambda_{12} = 0, \\ 0,5 \lambda_{11} - 3 \lambda_{12} = 0; \end{cases} \\ \begin{cases} 1,5 \lambda_{21} + 3 \lambda_{22} = 0, \\ 1 + 0,5 \lambda_{21} - 3 \lambda_{22} = 0, \end{cases} \end{cases}$$

найдем $\lambda_{11} = -\frac{1}{2}$, $\lambda_{12} = -\frac{1}{12}$, $\lambda_{21} = -\frac{1}{2}$, $\lambda_{22} = \frac{1}{4}$.

Условие $\lambda_{11}\lambda_{22} - \lambda_{12}\lambda_{21} \neq 0$ выполнено. Приведенная система имеет следующий вид:

$$\begin{cases} x_1 = g_1(x_1, x_2) = x_1 - \frac{1}{2}[x_2(x_1 - 1) - 1] - \frac{1}{12}[x_1^2 - x_2^2 - 1], \\ x_2 = g_2(x_1, x_2) = x_2 - \frac{1}{2}[x_2(x_1 - 1) - 1] + \frac{1}{4}[x_1^2 - x_2^2 - 1]. \end{cases} \quad (31)$$

Используя полученные представления (31) для функций $g_1(x_1, x_2)$ и $g_2(x_1, x_2)$, найдем векторы последовательных приближений по формулам (21). Оценку погрешности каждого приближения будем определять расстоянием между векторами двух последовательных итераций по n -норме

$$d_k = \|x^{(k)} - x^{(k-1)}\| = \max \{ |x_1^{(k)} - x_1^{(k-1)}|, |x_2^{(k)} - x_2^{(k-1)}| \}.$$

Тогда получим

$$x^{(1)} = \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \end{bmatrix} = \begin{bmatrix} g_1(x_1^{(0)}, x_2^{(0)}) \\ g_2(x_1^{(0)}, x_2^{(0)}) \end{bmatrix} = \begin{bmatrix} g_1(1,5; 1,5) \\ g_2(1,5; 1,5) \end{bmatrix} = \begin{bmatrix} 1,70833 \\ 1,37500 \end{bmatrix},$$

$$d_1 = 0,20833;$$

$$\mathbf{x}^{(2)} = \begin{bmatrix} x_1^{(2)} \\ x_2^{(2)} \end{bmatrix} = \begin{bmatrix} g_1(x_1^{(1)}, x_2^{(1)}) \\ g_2(x_1^{(1)}, x_2^{(1)}) \end{bmatrix} = \begin{bmatrix} g_1(1,70833; 1,37500) \\ g_2(1,70833; 1,37500) \end{bmatrix} = \begin{bmatrix} 1,71904 \\ 1,39497 \end{bmatrix},$$

$$d_2 = 0,01997;$$

$$\mathbf{x}^{(3)} = \begin{bmatrix} x_1^{(3)} \\ x_2^{(3)} \end{bmatrix} = \begin{bmatrix} g_1(x_1^{(2)}, x_2^{(2)}) \\ g_2(x_1^{(2)}, x_2^{(2)}) \end{bmatrix} = \begin{bmatrix} g_1(1,71904; 1,39497) \\ g_2(1,71904; 1,39497) \end{bmatrix} = \begin{bmatrix} 1,71676 \\ 1,39574 \end{bmatrix},$$

$$d_3 = 0,00281;$$

$$\mathbf{x}^{(4)} = \begin{bmatrix} x_1^{(4)} \\ x_2^{(4)} \end{bmatrix} = \begin{bmatrix} g_1(x_1^{(3)}, x_2^{(3)}) \\ g_2(x_1^{(3)}, x_2^{(3)}) \end{bmatrix} = \begin{bmatrix} g_1(1,71676; 1,39574) \\ g_2(1,71676; 1,39574) \end{bmatrix} = \begin{bmatrix} 1,71662 \\ 1,39533 \end{bmatrix},$$

$$d_4 = 0,00041;$$

$$\mathbf{x}^{(5)} = \begin{bmatrix} x_1^{(5)} \\ x_2^{(5)} \end{bmatrix} = \begin{bmatrix} g_1(x_1^{(4)}, x_2^{(4)}) \\ g_2(x_1^{(4)}, x_2^{(4)}) \end{bmatrix} = \begin{bmatrix} g_1(1,71662; 1,39533) \\ g_2(1,71662; 1,39533) \end{bmatrix} = \begin{bmatrix} 1,71667 \\ 1,39533 \end{bmatrix}.$$

Имеем $\|\mathbf{x}^{(5)} - \mathbf{x}^{(4)}\| = 0,00005$. Так как по условию $\varepsilon = 10^{-3}$, то согласно оценке (25б) можно взять в качестве решения 5-е

приближение $\xi = \mathbf{x}^{(5)} = \begin{bmatrix} 1,7167 \\ 1,3953 \end{bmatrix}$. ■

Упражнения

Методом итераций решить системы уравнений с точностью $\varepsilon = 10^{-2}$.

1. $\begin{cases} x_1^{2/3} + x_2^{2/3} = 4, \\ x_1^2 - 2x_2 = 0 \end{cases} \quad (x_1 > 0).$ 2. $\begin{cases} x_1^{2/3} + x_2^{2/3} = 4, \\ x_1^2 - 2x_2 = 0 \end{cases} \quad (x_1 < 0).$

3. $\begin{cases} x_2 - \sqrt{x_1 + 1} = 0, \\ x_1^2 + x_2^2 - 2x_2 = 0 \end{cases} \quad (x_1 > 0).$ 4. $\begin{cases} x_1 \cos x_1 - x_2 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 > 0).$

5. $\begin{cases} x_1 \cos x_1 - x_2 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 < 0).$ 6. $\begin{cases} 2x_1^2 + x_2^2 = 1, \\ x_2 - x_1^{2/3} = 0 \end{cases} \quad (x_2 > 0).$

7. $\begin{cases} 2x_1^2 + x_2^2 = 1, \\ x_2 + x_1^{2/3} = 0 \end{cases} \quad (x_2 < 0).$ 8. $\begin{cases} x_2 - \sin x_1 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 > 0).$

9. $\begin{cases} x_2 - \sin x_1 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 < 0).$
10. $\begin{cases} x_1^2 + x_2^2 - 2x_2 = 0, \\ x_2 - e^{-x_1} = 0 \end{cases} \quad (x_1 < 0).$
11. $\begin{cases} x_1^2 + x_2^2 - 2x_2 = 0, \\ x_2 - e^{-x_1} = 0 \end{cases} \quad (x_1 > 0).$
12. $\begin{cases} x_1^2 - 2x_1 + x_2^2 - 2x_2 + 1 = 0, \\ x_2 - \sqrt{x_1 + 1} = 0 \end{cases} \quad (x_1 > 0).$
13. $\begin{cases} x_1^2 + x_2^2 - 2x_1 = 0, \\ x_2 - \ln x_1 = 0 \end{cases} \quad (x_2 > 0).$
14. $\begin{cases} x_1^2 + 2x_1 + x_2^2 + 2x_2 + 1 = 0, \\ x_2 - \sqrt{x_1 + 1} = -1. \end{cases}$
15. $\begin{cases} x_1^{2/3} + x_2^{2/3} = 1, \\ x_1^2 + x_2^2 - 2x_1 = 0 \end{cases} \quad (x_2 > 0).$
16. $\begin{cases} x_1^{2/3} + x_2^{2/3} = 1, \\ x_2^2 + x_1^2 - 2x_1 = 0 \end{cases} \quad (x_2 < 0).$
17. $\begin{cases} x_1 \sin x_1 - x_2 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 > 0).$
18. $\begin{cases} x_1 \sin x_1 - x_2 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 < 0).$
19. $\begin{cases} \frac{x_1}{1+x_1^2} - x_2 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 > 0).$
20. $\begin{cases} \frac{x_1}{1+x_1^2} - x_2 = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 < 0).$
21. $\begin{cases} x_1^2 + x_2^2 - 2x_2 = 0, \\ x_2 - \frac{1}{2} \ln(x_1 + 1) = 0 \end{cases} \quad (x_1 > 0).$
22. $\begin{cases} x_1^2 + x_2^2 - 2x_2 = 0, \\ x_2 - 2 \ln(x_1 + 1) = 0 \end{cases} \quad (x_1 > 0).$
23. $\begin{cases} x_2 - 2x_1 e^{-x_1} = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 > 0).$
24. $\begin{cases} x_2 - 2x_1 e^{-x_1} = 0, \\ x_1^2 + x_2^2 - 1 = 0 \end{cases} \quad (x_1 < 0).$
25. $\begin{cases} (x_1^2 + x_2^2)^2 = 2(x_1^2 - x_2^2), \\ x_1^2 + x_2 - 1 = 0 \end{cases} \quad (x_1 > 0, \quad x_2 > 0).$
26. $\begin{cases} (x_1^2 + x_2^2)^2 = 2(x_1^2 - x_2^2), \\ x_1^2 + x_2 - 1 = 0 \end{cases} \quad (x_1 < 0, \quad x_2 > 0).$

Примечание. Для изображения кривой $(x_1^2 + x_2^2)^2 = 2(x_1^2 - x_2^2)$ (лемнискаты Бернулли) воспользоваться полярными координатами.

СРЕДНЕКВАДРАТИЧНОЕ ПРИБЛИЖЕНИЕ ФУНКЦИИ

§1. ИНТЕГРАЛЬНОЕ СРЕДНЕКВАДРАТИЧНОЕ ПРИБЛИЖЕНИЕ

ФУНКЦИИ ОРТОГОНАЛЬНЫМИ МНОГОЧЛЕНАМИ

Пусть на отрезке $[a, b]$ задана функция $f(x)$ и определена система функций $\{g_k(x)\}$ ($k=0, 1, 2, \dots$).

Обобщенным многочленом (полиномом) порядка n (степени n) относительно системы функций $\{g_k(x)\}$ называют функцию вида

$$Q_n(x) = C_0 g_0(x) + C_1 g_1(x) + \dots + C_n g_n(x) = \sum_{k=0}^n C_k g_k(x),$$

где $C_0, C_1, \dots, C_n$ — некоторые постоянные.

Обобщенный многочлен $Q_n(x)$ называют **многочленом наилучшего среднеквадратичного приближения функции $f(x)$ на отрезке $[a, b]$** , если расстояние от многочлена до функции $f(x)$ по среднеквадратичной норме (среднеквадратичное отклонение) наименьшее, т.е.

$$d(f, Q_n) = \left[\int_a^b (f(x) - Q_n(x))^2 dx \right]^{1/2} - \text{наименьшее.} \quad (1)$$

Задачу о нахождении такого многочлена $Q_n(x)$ называют **задачей об интегральном среднеквадратичном приближении** (аппроксимации) функции $f(x)$ на отрезке $[a, b]$ обобщенным многочленом. Эта задача сводится к нахождению коэффициентов $C_0, C_1, \dots, C_n$ из условия (1).

Если функция $f(x)$ и система функций $\{g_k(x)\}$ определены на множестве точек $\{x_1, \dots, x_m\}$, то приближение функции $f(x)$ многочленом $Q_n(x)$ называют **точечным среднеквадратичным приближением на множестве точек $\{x_1, \dots, x_m\}$** . При этом для обобщенного многочлена наилучшего приближения $Q_n(x)$ среднеквадратичное отклонение

$$d(f, Q_n) = \left[\sum_{i=1}^m (f(x_i) - Q_n(x_i))^2 \right]^{1/2}$$

на системе точек будет наименьшим.

Задача нахождения многочлена наилучшего приближения $Q_n(x)$ функции $f(x)$ на $[a, b]$ упрощается, если система функций $\{g_k(x)\}$

обладает свойством ортогональности на отрезке $[a, b]$. Предварительно рассмотрим некоторые определения.

Скалярным произведением функций $g_i(x)$ и $g_j(x)$ на отрезке $[a, b]$ называется интеграл от их произведения на этом отрезке. Обозначим скалярное произведение как (g_i, g_j) и запишем

$$(g_i, g_j) = \int_a^b g_i(x) g_j(x) dx.$$

Число $\|g_i\| = \sqrt{(g_i, g_i)} = \left[\int_a^b g_i^2(x) dx \right]^{1/2}$ является нормой функции $g_i(x)$ на отрезке $[a, b]$, а функция $f(x)$, для которой

существует интеграл $\int_a^b f^2(x) dx$, называется **интегрируемой с квадратом** на отрезке $[a, b]$.

Функции $g_i(x)$ и $g_j(x)$ называются **ортогональными** на отрезке $[a, b]$, если их скалярное произведение на этом отрезке равно нулю, т.е. если

$$(g_i, g_j) = \int_a^b g_i(x) g_j(x) dx = 0 \quad (i \neq j).$$

Система функций $\{g_k(x)\}$ ($k=0, 1, 2, \dots$) называется **ортогональной** на отрезке $[a, b]$, если все функции этой системы попарно ортогональны на этом отрезке.

Коэффициенты $C_0, C_1, \dots, C_n$ обобщенного многочлена

$$Q_n(x) = C_0 g_0(x) + C_1 g_1(x) + \dots + C_n g_n(x) \quad (2)$$

называются **коэффициентами Фурье** функции $f(x)$ относительно ортогональной системы функций, если они определяются по формулам

$$C_k = \frac{(f, g_k)}{\|g_k\|^2} = \frac{\int_a^b f(x) g_k(x) dx}{\int_a^b g_k^2(x) dx} \quad (k=0, 1, 2, \dots, n). \quad (3)$$

Теорема 1 (об интегральном среднеквадратичном приближении функции ортогональными многочленами). Для любой функции $f(x)$, интегрируемой с квадратом на отрезке $[a, b]$, обобщенный многочлен n -го порядка $Q_n(x)$ с коэффициентами Фурье функции $f(x)$ относительно ортогональной на отрезке $[a, b]$ системы функций $\{g_k(x)\}$ является многочленом наилучшего среднеквадратичного приближения этой функции, причем квадрат наименьшего среднеквадратичного отклонения определяется соотношением

$$\alpha^2(f, Q_n) = \|f\|^2 - \sum_{k=0}^n G_k^2 \|g_k\|^2, \quad (4)$$

где G_k — коэффициенты Фурье, определяемые по формулам (3).

Из формулы (4) видно, что с увеличением порядка обобщенного многочлена среднеквадратичное отклонение не увеличивается.

Замечание. Дадим геометрическую интерпретацию предыдущей теоремы и соотношения (4). Можно показать, что квадрат нормы обобщенного многочлена $\|Q_n\|^2$ относительно ортогональной системы функций $\{g_k(x)\}$ вычисляется по формуле

$$\|Q_n\|^2 = \sum_{k=0}^n G_k^2 \|g_k\|^2.$$

Запишем соотношение (4) в виде

$$\|f\|^2 = \|Q_n\|^2 + \alpha^2(f, Q_n).$$

Тогда это соотношение будем интерпретировать как формулировку теоремы Пифагора для прямоугольного треугольника с катетами, равными норме обобщенного многочлена наилучшего среднеквадратичного приближения $\|Q_n\|$ и величине отклонения $d(f, Q_n)$, и с гипотенузой, длина которой равна норме функции $\|f\|$ (рис. 15).

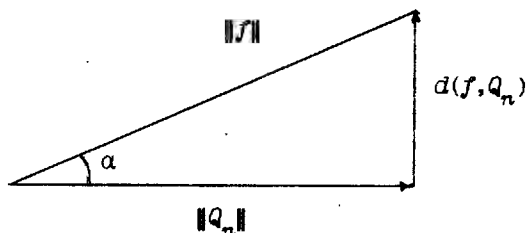


Рис. 15

§2. СРЕДНЕКВАДРАТИЧНОЕ ПРИБЛИЖЕНИЕ ФУНКЦИЙ ТРИГОНОМЕТРИЧЕСКИМИ МНОГОЧЛЕНАМИ

Пусть задана система тригонометрических функций

$$1, \cos \frac{\pi x}{l}, \sin \frac{\pi x}{l}, \cos \frac{2\pi x}{l}, \sin \frac{2\pi x}{l}, \dots, \cos \frac{k\pi x}{l}, \sin \frac{k\pi x}{l}, \dots \quad (5)$$

на отрезке $[-l, l]$.

Тригонометрическим многочленом n -й степени называют обобщенный многочлен по системе тригонометрических функций, имеющий вид

$$Q_n(x) = \frac{C_0}{2} + \sum_{k=1}^n \left[C_k \cos \frac{k\pi x}{l} + D_k \sin \frac{k\pi x}{l} \right], \quad (6)$$

где $C_0, C_1, \dots, C_n, D_1, D_2, \dots, D_n$ — некоторые числа.

Можно показать, что система тригонометрических функций (5) ортогональна на отрезке $[-l, l]$.

На основании теоремы 1 для функции $f(x)$, интегрируемой с квадратом на отрезке $[-l, l]$, тригонометрическим многочленом наилучшего среднеквадратичного приближения является тригонометрический многочлен

$$Q_n(x) = \frac{a_0}{2} + \sum_{k=1}^n \left[a_k \cos \frac{k\pi x}{l} + b_k \sin \frac{k\pi x}{l} \right], \quad (7)$$

где коэффициенты Фурье по тригонометрической системе функций для функции $f(x)$ определяются формулами (3) и имеют вид

$$a_0 = \frac{1}{l} \int_{-l}^l f(x) dx, \quad a_k = \frac{1}{l} \int_{-l}^l f(x) \cos \frac{k\pi x}{l} dx, \quad b_k = \frac{1}{l} \int_{-l}^l f(x) \sin \frac{k\pi x}{l} dx$$

$$(k = 1, 2, \dots). \quad (8)$$

Среднеквадратичное отклонение аппроксимирующего многочлена от функции $f(x)$, вычисляемое по формуле (4), в данном случае выражается равенством

$$d(f, Q_n) = \left[\int_{-l}^l (f(x) - Q_n(x))^2 dx \right]^{1/2} =$$

$$= \left[\int_{-l}^l f^2(x) dx - l \left(\frac{a_0^2}{2} + \sum_{k=1}^n (a_k^2 + b_k^2) \right) \right]^{1/2}. \quad (9)$$

Среднеквадратичное отклонение, отнесенное к норме аппроксимируемой функции $\|f\|$, характеризует точность приближения и обозначается $\bar{d}(f, Q_n) = d(f, Q_n) / \|f\|$ (величина $\sin \alpha$ на рис.15).

В частном случае, когда $f(x)$ — четная функция на отрезке $[-l, l]$, тригонометрический многочлен наилучшего среднеквадратичного приближения записывается в виде

$$Q_n(x) = \frac{a_0}{2} + \sum_{k=1}^n a_k \cos \frac{k\pi x}{l},$$

где коэффициенты Фурье

$$a_0 = \frac{2}{l} \int_0^l f(x) dx, \quad a_k = \frac{2}{l} \int_0^l f(x) \cos \frac{k\pi x}{l} dx \quad (k=1, 2, \dots, n).$$

Для нечетной функции $f(x)$ формулы (7) и (8) примут вид

$$Q_n(x) = \sum_{k=1}^n b_k \sin \frac{k\pi x}{l},$$

$$b_k = \frac{2}{l} \int_0^l f(x) \sin \frac{k\pi x}{l} dx \quad (k=1, 2, \dots, n).$$

Замечание. Тригонометрический многочлен (7) с коэффициентами Фурье (8) представляет собой n -ю частичную сумму ряда Фурье функции $f(x)$:

$$f(x) \sim \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_k \cos \frac{k\pi x}{l} + b_k \sin \frac{k\pi x}{l} \right). \quad (10)$$

Можно показать, что ряд Фурье сходится к функции $f(x)$ в среднеквадратичном. Точнее, для всякой функции $f(x)$, интегрируемой с квадратом на отрезке $[-l, l]$, имеет место предельное соотношение $\lim_{n \rightarrow \infty} d(f, Q_n) = 0$, определяющее сходимость последовательности частичных сумм $Q_n(x)$ к функции $f(x)$ по среднеквадратичной норме.

Важно отметить случай поточечной сходимости ряда Фурье к кусочно-гладкой функции, его порождающей.

Определение. Функция $f(x)$ называется **гладкой** на отрезке $[a, b]$, если на этом отрезке она непрерывна вместе со своей первой производной.

Функция $f(x)$ называется **кусочно-гладкой** на отрезке $[-1, 1]$, если этот отрезок можно разбить на конечное число отрезков, в каждом из которых $f(x)$ — гладкая функция.

Кусочно-гладкая функция может иметь конечное число разрывов первого рода. Предположим, что $x_0 \in (-1, 1)$ и является точкой разрыва. Обозначим левосторонний и правосторонний пределы функции $f(x)$ в точке x_0 следующим образом:

$$f(x_0 - 0) = \lim_{\substack{x \rightarrow x_0 \\ (x < x_0)}} f(x), \quad f(x_0 + 0) = \lim_{\substack{x \rightarrow x_0 \\ (x > x_0)}} f(x).$$

Естественно, что если функция непрерывна в точке x_0 , то

$$f(x_0 - 0) = f(x_0 + 0) = f(x_0).$$

Теорема 2. Тригонометрический ряд Фурье кусочно-гладкой функции $f(x)$ на отрезке $[-1, 1]$ сходится к значению

$$\frac{f(x_0 - 0) + f(x_0 + 0)}{2}$$

в точке x_0 (сходится к $f(x_0)$ в точках непрерывности функции). В обеих граничных точках сумма ряда равна среднему арифметическому ее предельных значений в этих точках, т.е.

$$\frac{f(-1 + 0) + f(1 - 0)}{2}.$$

Пример 1. Найти ряд Фурье для функции

$$f(x) = \begin{cases} 1, & -1 \leq x < 0; \\ x, & 0 < x \leq 1. \end{cases}$$

Представить графически приближение этой функции с помощью тригонометрических многочленов степеней $n_1 = 1$, $n_2 = 3$. Оценить погрешность среднеквадратичного приближения $\bar{d}(f, q_3)$.

Решение. Определяем коэффициенты Фурье:

$$a_0 = \int_{-1}^1 f(x) dx = \int_{-1}^0 dx + \int_0^1 x dx = \frac{3}{2};$$

$$\begin{aligned}
 a_k &= \int_{-1}^1 f(x) \cos \pi k x \, dx = \int_{-1}^0 \cos \pi k x \, dx + \int_0^1 x \cos \pi k x \, dx = \\
 &= \frac{1}{\pi k} \sin \pi k x \Big|_{-1}^0 + \frac{1}{\pi k} x \sin \pi k x \Big|_0^1 - \frac{1}{\pi k} \int_0^1 \sin \pi k x \, dx = \\
 &= \frac{1}{(\pi k)^2} \cos \pi k x \Big|_0^1 = \frac{1}{(\pi k)^2} (\cos \pi k - 1) = \begin{cases} 0, & k=2m, \\ -\frac{2}{\pi^2 (2m-1)^2}, & k=2m-1; \end{cases}
 \end{aligned}$$

$$\begin{aligned}
 b_k &= \int_{-1}^1 f(x) \sin \pi k x \, dx = \int_{-1}^0 \sin \pi k x \, dx + \int_0^1 x \sin \pi k x \, dx = \\
 &= -\frac{1}{\pi k} \cos \pi k x \Big|_{-1}^0 - \frac{1}{\pi k} x \cos \pi k x \Big|_0^1 + \frac{1}{\pi k} \int_0^1 \cos \pi k x \, dx = \\
 &= -\frac{1}{\pi k} (1 - \cos \pi k) - \frac{1}{\pi k} \cos \pi k + \frac{1}{(\pi k)^2} \sin \pi k x \Big|_0^1 = -\frac{1}{\pi k}.
 \end{aligned}$$

Таким образом, ряд Фурье имеет вид

$$\begin{aligned}
 f(x) &= \frac{3}{4} - \frac{2}{\pi^2} \sum_{m=1}^{\infty} \frac{\cos \pi (2m-1) x}{(2m-1)^2} - \frac{1}{\pi} \sum_{k=1}^{\infty} \frac{\sin \pi k x}{k} = \\
 &= \frac{3}{4} - \frac{2}{\pi^2} \left[\frac{\cos \pi x}{1^2} + \frac{\cos 3\pi x}{3^2} + \frac{\cos 5\pi x}{5^2} + \dots \right] - \\
 &\quad - \frac{1}{\pi} \left[\frac{\sin \pi x}{1} + \frac{\sin 2\pi x}{2} + \frac{\sin 3\pi x}{3} + \dots \right]. \quad (11)
 \end{aligned}$$

Многочлены наилучшего среднеквадратичного приближения получим как частичные суммы ряда (11):

$$Q_0(x) = \frac{3}{4}, \quad Q_1(x) = \frac{3}{4} - \frac{2}{\pi^2} \cos \pi x - \frac{\sin \pi x}{\pi},$$

$$Q_2(x) = \frac{3}{4} - \frac{2}{\pi^2} \cos \pi x - \frac{1}{\pi} \left[\sin \pi x + \frac{\sin 2\pi x}{2} \right],$$

$$Q_3(x) = \frac{3}{4} - \frac{2}{\pi^2} \left[\cos \pi x + \frac{\cos 3\pi x}{9} \right] - \frac{1}{\pi} \left[\sin \pi x + \frac{\sin 2\pi x}{2} + \frac{\sin 3\pi x}{3} \right].$$

Погрешность среднеквадратичного приближения находим по формуле (9):

$$d(f, Q_3) = \left\{ \int_{-1}^1 f^2(x) dx - \left[\frac{9}{8} + \frac{4}{\pi^4} \left[1 + \frac{1}{81} \right] + \frac{1}{\pi^2} \left[1 + \frac{1}{4} + \frac{1}{9} \right] \right] \right\}^{1/2} \approx$$

$$\approx 0,1699. \text{ Так как } \int_{-1}^1 f^2(x) dx = 1 + \int_0^1 x^2 dx = \frac{4}{3}, \text{ то } d(f, Q_3) = 0,147.$$

Графики функций $f(x)$, $Q_1(x)$ и $Q_3(x)$ изображены на рис.16. ■

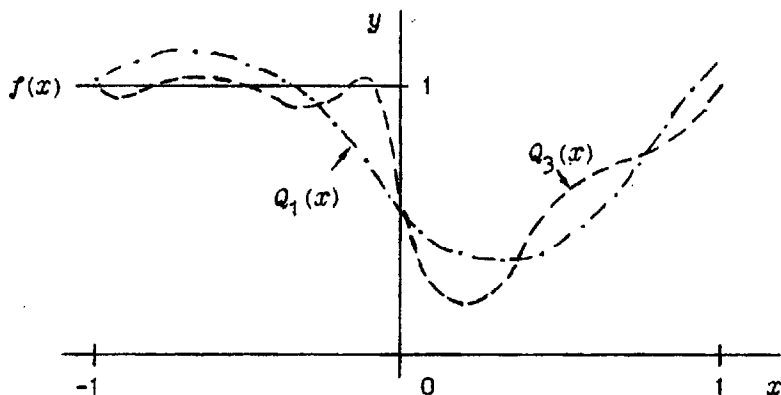


Рис.16

Пример 2. Функция

$$f(x) = \begin{cases} x, & 0 \leq x \leq 1; \\ 2-x, & 1 \leq x \leq 2 \end{cases}$$

разложить в ряд Фурье по синусам. Представить графически приближения этой функции с помощью тригонометрических многочленов степеней $n_1=1$, $n_2=5$. Оценить погрешности среднеквадратичного приближения $d(f, Q_5)$ и $\bar{d}(f, Q_5)$.

Решение. Функцию $f(x)$ доопределим нечетным образом на промежутке $[-2, 0]$. Полученную функцию $f^*(x)$ разложим в ряд Фурье (график функции $f^*(x)$ изображен на рис.17). Имеем

$$f^*(x) = \sum_{k=1}^{\infty} b_k \sin \frac{k\pi x}{2};$$

$$\begin{aligned} b_k &= \int_0^2 f(x) \sin \frac{k\pi x}{2} dx = \int_0^1 x \sin \frac{k\pi x}{2} dx + \int_1^2 (2-x) \sin \frac{k\pi x}{2} dx = \\ &= -\frac{2}{\pi k} x \cos \frac{\pi k x}{2} \Big|_0^1 + \frac{2}{\pi k} \int_0^1 \cos \frac{\pi k x}{2} dx - \frac{2}{\pi k} (2-x) \cos \frac{\pi k x}{2} \Big|_1^2 - \\ &- \frac{2}{\pi k} \int_1^2 \cos \frac{\pi k x}{2} dx = -\frac{2}{\pi k} \cos \frac{\pi k}{2} + \frac{4}{(\pi k)^2} \sin \frac{\pi k}{2} \Big|_0^1 + \frac{2}{\pi k} \cos \frac{\pi k}{2} - \end{aligned}$$

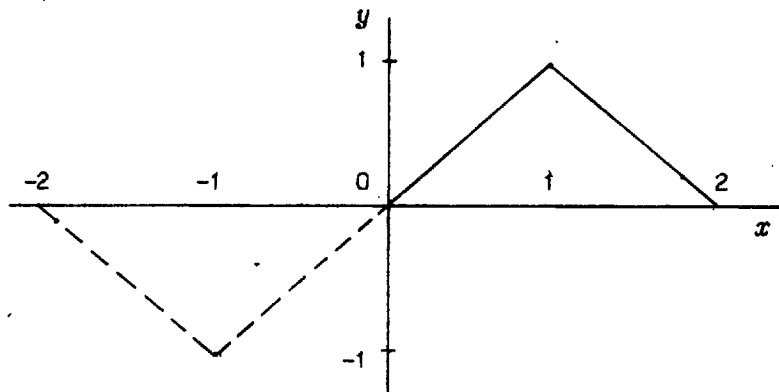


Рис.17

$$\begin{aligned}
& -\frac{4}{(\pi k)^2} \sin \frac{\pi k x}{2} \Big|_1^2 = \frac{8}{(\pi k)^2} \sin \frac{\pi k}{2} = \\
& = \begin{cases} 0, & k=2m; \\ (-1)^{m+1} \frac{8}{\pi(2m-1)^2}, & k=2m-1. \end{cases}
\end{aligned}$$

Ряд Фурье функции $f^*(x)$ имеет вид

$$f^*(x) = -\frac{8}{\pi^2} \sum_{m=1}^{\infty} \frac{(-1)^m}{(2m-1)^2} \sin \frac{\pi(2m-1)x}{2}. \quad (12)$$

Этот ряд сходится поточечно к функции $f(x)$ на отрезке $[0, 2]$. Частичные суммы ряда (12) являются многочленами наилучшего среднеквадратичного приближения функции $f(x)$ на отрезке $[0, 2]$:

$$\begin{aligned}
Q_1(x) &= \frac{8}{\pi^2} \sin \frac{\pi x}{2}, \\
Q_3(x) &= \frac{8}{\pi^2} \left(\sin \frac{\pi x}{2} - \frac{1}{9} \sin \frac{3\pi x}{2} \right), \\
Q_5(x) &= \frac{8}{\pi^2} \left(\sin \frac{\pi x}{2} - \frac{1}{9} \sin \frac{3\pi x}{2} + \frac{1}{25} \sin \frac{5\pi x}{2} \right), \\
&\dots \dots \dots
\end{aligned}$$

Найдем погрешность среднеквадратичного приближения многочленом $Q_5(x)$:

$$d(f, Q_5) = \left[\int_0^2 f^2(x) dx - \frac{64}{\pi^4} \left[1 + \frac{1}{81} + \frac{1}{625} \right] \right]^{1/2} \approx 0,0219.$$

Далее, учитывая, что $\int_0^2 f^2(x) dx = \int_0^1 x^2 dx + \int_1^2 (2-x)^2 dx = \frac{2}{3}$, по-

лучим $\bar{d}(f, Q_5) = 0,0269$.

Графики функций $f(x)$, $Q_1(x)$ и $Q_5(x)$ изображены на рис.18. ■

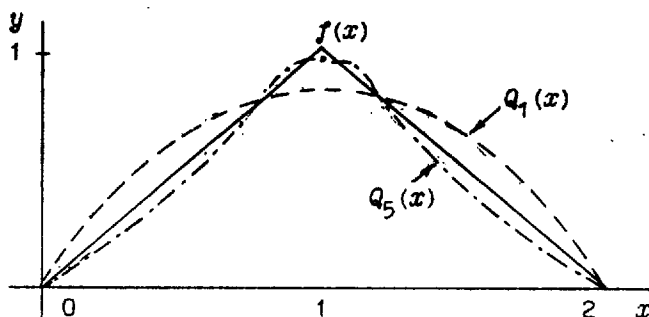


Рис. 18

Упражнения

Найти ряд Фурье для функции $f(x)$, представить графически приближения этой функции с помощью тригонометрических многочленов степеней n_1, n_2, n_3 . Оценить погрешность среднеквадратичного приближения $\bar{\sigma}(f, Q_{n_3})$.

$$1. f(x) = \begin{cases} 1, & -1 \leq x < 0; \\ 2, & 0 < x \leq 1 \end{cases}$$

$$(n_1=3, n_2=5, n_3=7).$$

$$3. f(x) = \begin{cases} x, & 0 \leq x \leq 1; \\ 2-x, & 1 \leq x \leq 2 \end{cases}$$

(разложить по косинусам)

$$(n_1=1, n_2=3, n_3=5).$$

$$5. f(x) = \begin{cases} -2x, & -\pi \leq x \leq 0; \\ 3x, & 0 \leq x \leq \pi \end{cases}$$

$$(n_1=1, n_2=3, n_3=5).$$

$$7. f(x) = \pi - 2x, \quad 0 \leq x \leq \pi$$

(разложить по косинусам)

$$(n_1=1, n_2=3, n_3=5).$$

$$9. f(x) = \pi + x, \quad -\pi \leq x \leq \pi$$

$$(n_1=3, n_2=5, n_3=7).$$

$$2. f(x) = 1-x, \quad x \in [0, 1]$$

(разложить по косинусам)

$$(n_1=1, n_2=3, n_3=5).$$

$$4. f(x) = \begin{cases} -x, & -\pi \leq x < 0; \\ 0, & 0 \leq x \leq \pi \end{cases}$$

$$(n_1=1, n_2=3, n_3=5).$$

$$6. f(x) = \begin{cases} -2, & -\pi \leq x < 0; \\ 2, & 0 < x \leq \pi \end{cases}$$

$$(n_1=3, n_2=5, n_3=7).$$

$$8. f(x) = \pi - 2x, \quad 0 < x \leq \pi$$

(разложить по синусам)

$$(n_1=2, n_2=4, n_3=6).$$

$$10. f(x) = \begin{cases} 2, & -\pi \leq x < 0; \\ -2, & 0 < x \leq \pi \end{cases}$$

$$(n_1=3, n_2=5, n_3=7).$$

$$11. f(x) = \begin{cases} 1, & -\pi \leq x < 0; \\ -2, & 0 < x \leq \pi \end{cases}$$

$$(n_1=3, n_2=5, n_3=7).$$

$$13. f(x) = \begin{cases} x, & -\pi \leq x \leq 0; \\ 2x, & 0 \leq x \leq \pi \end{cases}$$

$$(n_1=1, n_2=3, n_3=5).$$

$$15. f(x) = 2+|x|, \quad -2 \leq x \leq 2$$

$$(n_1=1, n_2=3, n_3=5).$$

$$17. f(x) = \begin{cases} 2x, & -\pi \leq x \leq 0; \\ -3x, & 0 \leq x \leq \pi \end{cases}$$

$$(n_1=1, n_2=3, n_3=5).$$

$$19. f(x) = \begin{cases} x, & -4 \leq x \leq 0; \\ 2x, & 0 \leq x \leq 4 \end{cases}$$

$$(n_1=1, n_2=3, n_3=5).$$

$$21. f(x) = \begin{cases} -1, & -1 \leq x < 0; \\ 3, & 0 < x \leq 1 \end{cases}$$

$$(n_1=3, n_2=5, n_3=7).$$

$$23. f(x) = \begin{cases} 1, & -1 \leq x \leq 0; \\ 1+x, & 0 \leq x \leq 1 \end{cases}$$

$$(n_1=2, n_2=3, n_3=4).$$

$$25. f(x) = \begin{cases} 0, & -2 \leq x < 0; \\ 2-x, & 0 < x \leq 2 \end{cases}$$

$$(n_1=3, n_2=4, n_3=5).$$

$$12. f(x) = \begin{cases} -\pi, & -\pi \leq x < 0; \\ x-\pi, & 0 \leq x \leq \pi \end{cases}$$

$$(n_1=1, n_2=3, n_3=5).$$

$$14. f(x) = 2x, \quad 0 \leq x \leq 1$$

(разложить по косинусам)

$$(n_1=1, n_2=3, n_3=5).$$

$$16. f(x) = 2-|x|, \quad -2 \leq x \leq 2$$

$$(n_1=1, n_2=3, n_3=5).$$

$$18. f(x) = \begin{cases} 0, & -2 \leq x \leq 0; \\ x, & 0 \leq x \leq 2 \end{cases}$$

$$(n_1=1, n_2=3, n_3=5).$$

$$20. f(x) = \begin{cases} -x, & -2 \leq x \leq 0; \\ 0, & 0 \leq x \leq 2 \end{cases}$$

$$(n_1=2, n_2=3, n_3=4).$$

$$22. f(x) = \begin{cases} 0, & -2 \leq x < 0; \\ -2, & 0 < x \leq 2 \end{cases}$$

$$(n_1=3, n_2=5, n_3=7).$$

$$24. f(x) = \begin{cases} 1+x, & -1 \leq x \leq 0; \\ 1, & 0 \leq x \leq 1 \end{cases}$$

$$(n_1=2, n_2=3, n_3=4).$$

§3. СРЕДНЕКВАДРАТИЧНОЕ ПРИБЛИЖЕНИЕ ФУНКЦИИ АЛГЕБРАИЧЕСКИМИ МНОГОЧЛЕНАМИ ЛЕЖАНДРА

Многочленами Лежандра называются алгебраические многочлены, определяемые следующими формулами:

$$P_0(x) \equiv 1, \quad P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} (x^2 - 1)^n \quad (n=1, 2, \dots). \quad (13)$$

Выполнив дифференцирование, найдем несколько первых многочленов:

$$P_1(x) = x, \quad P_2(x) = \frac{1}{2}(3x^2 - 1), \quad P_3(x) = \frac{1}{2}(5x^3 - 3x), \quad (14)$$

$$P_4(x) = \frac{1}{8}(35x^4 - 30x^2 + 3), \quad P_5(x) = \frac{1}{8}(63x^5 - 70x^3 + 15x),$$

Обобщенный многочлен степени n относительно системы алгебраических многочленов Лежандра $\{P_n(x)\}$ имеет вид

$$Q_n(x) = C_0 P_0(x) + C_1 P_1(x) + \dots + C_n P_n(x) = \sum_{k=0}^n C_k P_k(x), \quad (15)$$

где $C_0, C_1, \dots, C_n$ — некоторые постоянные.

Система многочленов Лежандра (13) ортогональна на отрезке $[-1, 1]$, т.е.

$$\int_{-1}^1 P_n(x) P_m(x) dx = 0 \quad (n \neq m, n=0, 1, \dots, m=0, 1, \dots).$$

Среднеквадратичную норму многочлена $P_n(x)$ можно определить по формуле

$$\|P_n\| = \left[\int_{-1}^1 P_n^2(x) dx \right]^{1/2} = \sqrt{\frac{2}{2n+1}}. \quad (16)$$

На основании теоремы 1 для функции $f(x)$, интегрируемой с квадратом на отрезке $[-1, 1]$, обобщенным многочленом наилучшего среднеквадратичного приближения по системе многочленов Лежандра является многочлен (15) с коэффициентами Фурье

$$C_k = \frac{(f, P_k)}{\|P_k\|^2} = \frac{\int_{-1}^1 f(x) P_k(x) dx}{\left[\sqrt{\frac{2}{2k+1}} \right]^2} = \frac{2k+1}{2} \int_{-1}^1 f(x) P_k(x) dx \quad (k=0, 1, 2, \dots). \quad (17)$$

Среднеквадратичное отклонение аппроксимирующего многочлена

от функции $f(x)$, вычисляемое по формуле (4), в данном случае определяется равенством

$$d(f, Q_n) = \left[\|f\|^2 - \sum_{k=0}^n \sigma_k^2 \|P_k\|^2 \right]^{1/2}. \quad (18)$$

Из формул (14) видно, что многочлены $P_k(x)$ четны при четном значении k и нечетны при нечетном k .

Для четной на отрезке $[-1, 1]$ функции $f(x)$ коэффициенты Фурье таковы:

$$C_{2k} = (4k + 1) \int_0^1 f(x) P_{2k}(x) dx, \quad C_{2k+1} = 0 \quad (k=0, 1, 2, \dots). \quad (19)$$

Аналогично, для нечетной функции $f(x)$ получаем

$$C_{2k} = 0, \quad C_{2k+1} = (4k + 3) \int_0^1 f(x) P_{2k+1}(x) dx \quad (k=0, 1, 2, \dots). \quad (20)$$

Замечание. Обобщенный многочлен (15) относительно многочленов Лежандра с коэффициентами Фурье (17) представляет собой n -ю частичную сумму ряда, называемого рядом Фурье - Лежандра:

$$f(x) \sim \sum_{k=0}^n C_k P_k(x). \quad (21)$$

Ряд Фурье - Лежандра сходится к функции $f(x)$ в среднеквадратичном на отрезке $[-1, 1]$. Точечная сходимость ряда Фурье - Лежандра к кусочно-гладкой функции, его порождающей, определяется следующей теоремой.

Теорема 3. Ряд Фурье - Лежандра кусочно-гладкой функции $f(x)$ на отрезке $[-1, 1]$ сходится к значению

$$\frac{f(x_0 - 0) + f(x_0 + 0)}{2}$$

в точке x_0 (сходится к $f(x_0)$ в точках непрерывности функции).

Замечание. Если функция $f(x)$ определена на отрезке $[a, b]$, то с помощью линейного преобразования

$$x = \frac{b-a}{2} t + \frac{b+a}{2}, \quad t \in [-1, 1], \quad (22)$$

можно получить многочлены Лежандра, ортогональные на отрезке $[a, b]$:

$$\tilde{P}_k(x) = P_k(t(x)) = P_k\left[\frac{x - \frac{b+a}{2}}{\frac{b-a}{2}}\right]. \quad (23)$$

Доказать ортогональность многочленов Лежандра $\tilde{P}_k(x)$ на отрезке $[a, b]$ и найти норму этих многочленов можно с помощью замены переменной под знаком интеграла. Действительно,

$$\int_a^b \tilde{P}_n(x) \tilde{P}_m(x) dx = \frac{b-a}{2} \int_{-1}^1 P_n(t) P_m(t) dt = 0, \\ (n \neq m, n=0, 1, \dots, m=0, 1, \dots).$$

При $m = n$ находим

$$\|\tilde{P}_n\| = \left[\int_a^b \tilde{P}_n^2(x) dx \right]^{1/2} = \sqrt{\frac{b-a}{2}} \|P_n\| = \sqrt{\frac{b-a}{2n+1}}.$$

Выражение для коэффициентов Фурье в обобщенном многочлене по многочленам Лежандра $\tilde{P}_k(x)$ для функции $f(x)$, определенной на отрезке $[a, b]$, примет вид

$$C_k = \frac{(f, \tilde{P}_k)}{\|\tilde{P}_k\|^2} = \frac{\int_a^b f(x) \tilde{P}_k(x) dx}{\left[\sqrt{\frac{b-a}{2k+1}} \right]^2} = \frac{2k+1}{b-a} \int_a^b f(x) \tilde{P}_k(x) dx \quad (k=0, 1, 2, \dots).$$

Произведя замену переменной по формуле (22), получим

$$C_k = \frac{2k+1}{2} \int_{-1}^1 \tilde{f}(t) P_k(t) dt \quad (k=0, 1, 2, \dots), \quad (24)$$

где

$$\tilde{f}(t) = f\left[\frac{b-a}{2} t + \frac{b+a}{2}\right].$$

Пример 1. Найти алгебраический многочлен степени $n=2$ наилучшего среднеквадратичного приближения для функции $f(x)=|1-x|$ ($0 \leq x \leq 2$). Оценить погрешность $\bar{d}(f, Q_2) = d(f, Q_2) / \|f\|$.

Решение. Имеем $f(x) \approx Q_2(x) = C_0 \tilde{P}_0(x) + C_1 \tilde{P}_1(x) + C_2 \tilde{P}_2(x)$.

Используя формулы (14), (22) и (23) и учитывая, что $a=0$ и $b=2$, находим

$$\begin{aligned} \tilde{P}_0(x) &= 1, & \tilde{P}_1(x) &= P_1 \left[\frac{x - \frac{b+a}{2}}{\frac{b-a}{2}} \right] = P_1(x-1) = x-1, \\ \tilde{P}_2(x) &= P_2(x-1) = \frac{1}{2} (3(x-1)^2 - 1) = \frac{1}{2} (3x^2 - 6x + 2). \end{aligned} \quad (25)$$

Производя замену переменной по формуле (22), получим

$$\tilde{f}(t) = f(x(t)) = |t| \quad (-1 \leq t \leq 1),$$

где коэффициенты Фурье вычисляются по формулам (24):

$$C_k = \frac{2k+1}{2} \int_{-1}^1 |t| P_k(t) dt \quad (k=0, 1, 2, \dots).$$

Так как функция $\tilde{f}(t) = |t|$ — четная, то можно воспользоваться формулами (19):

$$C_{2k} = (4k+1) \int_0^1 |t| P_{2k}(t) dt, \quad C_{2k+1} = 0 \quad (k=0, 1, 2, \dots).$$

Следовательно,

$$C_0 = \int_0^1 t P_0(t) dt = \int_0^1 t dt = \frac{1}{2}, \quad C_1 = 0,$$

$$C_2 = 5 \int_0^1 t P_2(t) dt = 5 \int_0^1 t \frac{1}{2} (3t^2 - 1) dt = \frac{5}{8},$$

откуда

$$\begin{aligned} Q_2(x) &= \frac{1}{2} \tilde{P}_0(x) + \frac{5}{8} \tilde{P}_2(x) = \frac{1}{2} + \frac{5}{8} \cdot \frac{1}{2} (3x^2 - 6x + 2) = \\ &= \frac{3}{16} (5x^2 - 10x + 6). \end{aligned}$$

По формуле (18) находим погрешность среднеквадратичного приближения:

$$d(f, Q_2) = \left(\|f\|^2 - \sum_{k=0}^2 C_k^2 \|\tilde{P}_k\|^2 \right)^{1/2} = \left[\int_0^2 |1-x|^2 dx - \left[\frac{1}{4} \cdot 2 + \frac{25}{64} \cdot \frac{2}{5} \right] \right]^{1/2},$$

т.е. $d(f, Q_2) = 0,102$, $\bar{d}(f, Q_2) = 0,125$. Графики функций $y = |1-x|$ и $Q_2(x)$ ($0 \leq x \leq 2$) изображены на рис.19. ■

Пример 2. Найти алгебраический многочлен степени $n=2$ наилучшего среднеквадратичного приближения для функции

$$f(x) = \begin{cases} 1, & 0 \leq x \leq 1; \\ 2-x, & 1 \leq x \leq 2. \end{cases}$$

Оценить погрешность $\bar{d}(f, Q_2)$.

Решение. Имеем $f(x) \approx Q_2(x) = C_0 \tilde{P}_0(x) + C_1 \tilde{P}_1(x) + C_2 \tilde{P}_2(x)$, где $\tilde{P}_0(x)$, $\tilde{P}_1(x)$, $\tilde{P}_2(x)$ определены соотношениями (23). Производя замену переменной по формуле (22), получим для дальнейших расчетов функцию $\tilde{f}(t)$, определенную на отрезке $[-1, 1]$. Так как $a=0$, $b=2$, $x = t+1$, то

$$\tilde{f}(t) = f(x(t)) = \begin{cases} 1, & -1 \leq t \leq 0; \\ 1-t, & 0 \leq t \leq 1. \end{cases}$$

Коэффициенты Фурье найдем по формулам (17):

$$C_0 = \frac{1}{2} \int_{-1}^1 \tilde{f}(t) P_0(t) dt = \frac{1}{2} \left[\int_{-1}^0 dt + \int_0^1 (1-t) dt \right] = \frac{3}{4}.$$

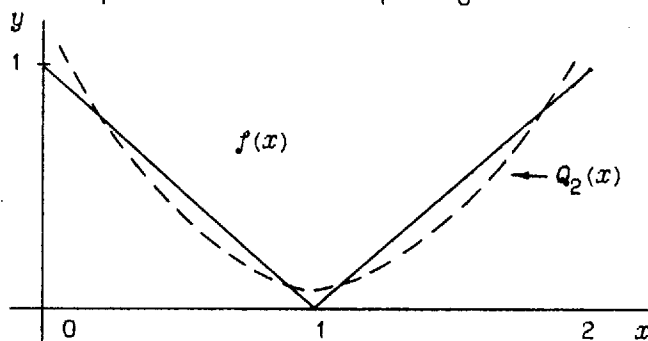


Рис.19

$$C_1 = \frac{3}{2} \int_{-1}^1 \tilde{f}(t) P_1(t) dt = \frac{3}{2} \left[\int_{-1}^0 t dt + \int_0^1 (1-t)t dt \right] = -\frac{1}{2},$$

$$C_2 = \frac{5}{2} \int_{-1}^1 \tilde{f}(t) P_2(t) dt = \frac{5}{2} \left[\int_{-1}^0 \frac{1}{2} (3t^2 - 1) dt + \right. \\ \left. + \int_0^1 (1-t) \frac{1}{2} (3t^2 - 1) dt \right] = -\frac{5}{16}.$$

Таким образом,

$$Q_2(x) = \frac{3}{4} \tilde{P}_0(x) - \frac{1}{2} \tilde{P}_1(x) - \frac{5}{16} \tilde{P}_2(x) = \\ = \frac{3}{4} - \frac{1}{2} (x-1) - \frac{5}{16} \cdot \frac{1}{2} (3x^2 - 6x + 2) = \frac{1}{32} (-15x^2 + 14x + 30).$$

По формуле (18) находим погрешность среднеквадратичного приближения:

$$d(f, Q_2) = \left[\|f\|^2 - \sum_{k=0}^2 C_k^2 \|\tilde{P}_k\|^2 \right]^{1/2}, \quad \bar{d}(f, Q_2) = d(f, Q_2) / \|f\|.$$

$$\|f\|^2 = \int_0^2 f^2(x) dx = \int_{-1}^1 \tilde{f}^2(t) dt = 1 + \int_0^1 (1-t)^2 dt = \frac{4}{3},$$

$$\|\tilde{P}_0\|^2 = 2, \quad \|\tilde{P}_1\|^2 = \frac{2}{3}, \quad \|\tilde{P}_2\|^2 = \frac{2}{5},$$

$$d(f, Q_2) = \left[\frac{4}{3} - \left[\frac{9}{8} + \frac{1}{6} + \frac{5}{128} \right] \right]^{1/2} \approx 0,051, \quad \bar{d}(f, Q_2) \approx 0,0442.$$

Графики функций $f(x)$ и $Q_2(x)$ ($0 \leq x \leq 2$) изображены на рис.20. ■

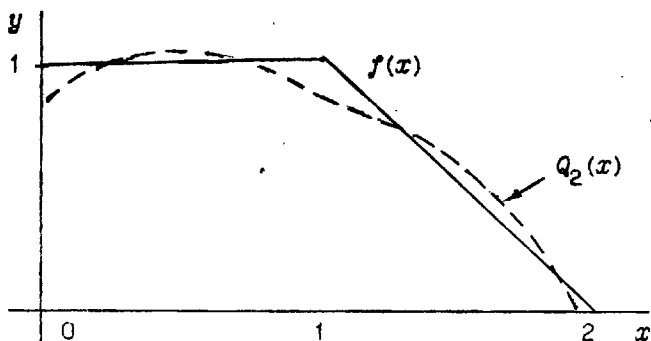


Рис. 20

Упражнения

Найти алгебраический многочлен наилучшего среднеквадратичного приближения $Q_n(x)$ для функции $f(x)$ на отрезке $[a, b]$. Представить графически приближение функции $f(x)$ многочленом $Q_n(x)$. Оценить погрешность среднеквадратичного приближения $\bar{d}(f, Q_n)$.

$$1. \quad f(x) = \begin{cases} -2, & -2 \leq x \leq 0; \\ x-2, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$2. \quad f(x) = \begin{cases} 2x, & -2 \leq x \leq 0; \\ -3x, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$3. \quad f(x) = \begin{cases} -x, & 0 \leq x \leq 1; \\ x-2, & 1 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$4. \quad f(x) = \begin{cases} x, & -2 \leq x \leq 0; \\ \frac{1}{2}x, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$5. \quad f(x) = \begin{cases} x, & -2 \leq x \leq 0; \\ -2x, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$6. \quad f(x) = \begin{cases} -x, & -2 \leq x \leq 0; \\ 2x, & 0 \leq x \leq 2 \end{cases} \\ (n = 3).$$

$$7. \quad f(x) = \begin{cases} x, & -2 \leq x \leq 0; \\ 2x, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$8. \quad f(x) = \begin{cases} x, & -3 \leq x \leq 0; \\ 0, & 0 \leq x \leq 3 \end{cases} \\ (n = 2).$$

$$9. \quad f(x) = \begin{cases} 1, & 0 \leq x \leq 1; \\ x, & 1 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$10. \quad f(x) = \begin{cases} 0, & -1 \leq x \leq 0; \\ x, & 0 \leq x \leq 1 \end{cases} \\ (n = 4).$$

$$11. \quad f(x) = \begin{cases} 1, & -1 \leq x \leq 0; \\ 1-x, & 0 \leq x \leq 1 \end{cases} \\ (n = 2).$$

$$12. \quad f(x) = \begin{cases} -2x, & -2 \leq x \leq 0; \\ 3x, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$13. \quad f(x) = \begin{cases} 2, & -2 \leq x \leq 0; \\ 2-x, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$14. \quad f(x) = \begin{cases} -x, & -2 \leq x \leq 0; \\ -\frac{1}{2}x, & 0 \leq x \leq 2 \end{cases} \\ (n = 2).$$

$$15. \quad f(x) = \begin{cases} 0, & -1 \leq x \leq 0; \\ -x, & 0 \leq x \leq 1 \end{cases} \\ (n = 4).$$

$$16. \quad f(x) = \begin{cases} 0, & -1 \leq x < 0; \\ 1, & 0 < x \leq 1 \end{cases} \\ (n = 5).$$

$$17. \quad f(x) = 2-|x|, \quad -2 \leq x \leq 2 \\ (n = 4).$$

$$18. \quad f(x) = \sqrt{|x-1|}, \quad 0 \leq x \leq 2 \\ (n = 4).$$

$$19. \quad f(x) = 1-|x|, \quad -1 \leq x \leq 1 \\ (n = 4).$$

$$20. \quad f(x) = 2|x|, \quad -2 \leq x \leq 2 \\ (n = 4).$$

$$21. \quad f(x) = -1+|x|, \quad -1 \leq x \leq 1 \\ (n = 4).$$

$$22. \quad f(x) = \begin{cases} 0, & -2 \leq x < 0; \\ 2, & 0 < x \leq 2 \end{cases} \\ (n = 5).$$

$$23. \quad f(x) = 1+|x|, \quad -1 \leq x \leq 1 \\ (n = 4).$$

$$24. \quad f(x) = \begin{cases} -1, & -1 \leq x < 0; \\ 1, & 0 < x \leq 1 \end{cases} \\ (n = 5).$$

$$25. \quad f(x) = |x|, \quad -1 \leq x \leq 1 \\ (n = 4).$$

§4. ТОЧЕЧНОЕ СРЕДНЕКВАДРАТИЧНОЕ ПРИБЛИЖЕНИЕ ФУНКЦИИ ОРТОГОНАЛЬНЫМИ МНОГОЧЛЕНАМИ. ОРТОГОНАЛЬНЫЕ МНОГОЧЛЕНЫ ЧЕБЫШЕВА

Пусть на множестве точек $\{x_i\}$ ($i=1,2,\dots,m$) задана функция $f(x)$ и определена система функций $\{g_k(x)\}$ ($k=0,1,2,\dots$).

Скалярным произведением функций $g_k(x)$ и $g_l(x)$ на множестве точек $\{x_i\}$ ($i=1,2,\dots,m$) называется сумма произведений значений функций, вычисленных во всех точках, т.е.

$$(g_k, g_l) = \sum_{i=1}^m g_k(x_i) g_l(x_i).$$

Число $\|g_k\| = \sqrt{(g_k, g_k)} = \left[\sum_{i=1}^m g_k^2(x_i) \right]^{1/2}$ является нормой функции $g_k(x)$ на множестве точек $\{x_i\}$ ($i=1,2,\dots,m$).

Функции $g_k(x)$ и $g_l(x)$ называются ортогональными на множестве точек, если их скалярное произведение на этом множестве равно нулю, т.е.

$$(g_k, g_l) = \sum_{i=1}^m g_k(x_i) g_l(x_i) = 0 \quad (k \neq l). \quad (26)$$

Система функций $\{g_k(x)\}$ ($k=0,1,2,\dots$) называется ортогональной на множестве точек $\{x_i\}$ ($i=1,2,\dots,m$), если все функции этой системы попарно ортогональны на этом множестве.

Коэффициенты $C_0, C_1, \dots, C_n$ обобщенного многочлена

$$Q_n(x) = C_0 g_0(x) + C_1 g_1(x) + \dots + C_n g_n(x)$$

называются коэффициентами Фурье функции $f(x)$ относительно ортогональной системы функций, если они определяются по формулам

$$C_k = \frac{(f, g_k)}{\|g_k\|^2} = \frac{\sum_{i=1}^m f(x_i) g_k(x_i)}{\sum_{i=1}^m g_k^2(x_i)} \quad (k=0,1,2,\dots,n). \quad (27)$$

Теорема 4 (о точечном среднеквадратичном приближении функции ортогональными многочленами). Для функции $f(x)$, определенной на множестве точек $\{x_i\}$ ($i=1,2,\dots,m$), обобщенный многочлен n -й

степени $Q_n(x)$ с коэффициентами Фурье относительно ортогональной на множестве точек системы функций $\{g_k(x)\}$ является многочленом наилучшего среднеквадратичного приближения этой функции, причем квадрат наименьшего среднеквадратичного отклонения определяется соотношением

$$\sigma^2(f, Q_n) = \|f\|^2 - \sum_{k=0}^n C_k^2 \|g_k\|^2, \quad (28)$$

где C_k — коэффициенты Фурье, определяемые по формулам (27). Оценка погрешности приближения определяется величиной

$$\bar{\sigma}(f, Q_n) = \sigma(f, Q_n) / \|f\|.$$

Многочленами Чебышева на множестве точек $\{x_i\}$ ($i=1, 2, \dots, m$) называются алгебраические многочлены, ортогональные на этом множестве, с нормой $\|g_k\|$, отличной от нуля, и определяемые следующими рекуррентными формулами:

$$g_0 = 1; \quad g_1(x) = x - a; \quad g_k(x) = (x - a_k)g_{k-1}(x) - b_k g_{k-2}(x) \quad (k=2, 3, \dots, m-1), \quad (29)$$

$$\text{где } a = -\frac{1}{m} \sum_{i=1}^m x_i, \quad a_k = \frac{\sum_{i=1}^m x_i g_{k-1}^2(x_i)}{\sum_{i=1}^m g_{k-1}^2(x_i)}, \quad b_k = \frac{\sum_{i=1}^m x_i g_{k-2}(x_i) g_{k-1}(x_i)}{\sum_{i=1}^m g_{k-2}^2(x_i)}.$$

Соотношения для коэффициентов a , a_k , b_k получены из условия ортогональности (26).

Замечание. Можно показать, что многочлен $g_m(x)$ степени m на множестве точек $\{x_i\}$ ($i=1, 2, \dots, m$), полученный по рекуррентным формулам (29), на этом множестве точек имеет норму, равную нулю, и уже не является многочленом Чебышева.

Пример 1. На множестве двух точек $\{0, 1\}$ определить ортогональные многочлены Чебышева и вычислить их нормы.

Решение. Здесь $m = 2$. Имеем

$$g_0(x) = 1, \quad g_1(x) = x - a, \quad a = \frac{1}{2} \sum_{i=1}^2 x_i = \frac{1}{2} (0+1) = \frac{1}{2},$$

$$\|g_0\|^2 = 2, \quad \|g_1\|^2 = g_1^2(0) + g_1^2(1) = \frac{1}{2},$$

$$a_2 = \frac{\sum_{i=1}^2 x_i g_1^2(x_i)}{\|g_1\|^2} = \frac{0 \cdot \frac{1}{4} + 1 \cdot \frac{1}{4}}{\|g_1\|^2} = \frac{1}{2},$$

$$b_2 = \frac{\sum_{i=1}^2 x_i g_0(x_i) g_1(x_i)}{\|g_0\|^2} = \frac{0 \cdot 1 \cdot \left[-\frac{1}{2}\right] + 1 \cdot 1 \cdot \frac{1}{2}}{2} = \frac{1}{4}.$$

Далее, используя рекуррентные соотношения (29), построим функцию

$$g_2(x) = (x - a_2)g_1(x) - b_2g_0(x) = \left[x - \frac{1}{2}\right] \left[x - \frac{1}{2}\right] - \frac{1}{4}.$$

Норма функции $g_2(x)$ на множестве $\{0, 1\}$ равна нулю и, следовательно, эта функция не является многочленом Чебышева. ■

Пример 2. Функция $y = f(x)$ определена таблицей вида

t	1	2	3	4
x_t	0	1	3	4
y_t	4	0	1	2

Требуется аппроксимировать функцию $y = f(x)$ алгебраическими многочленами наилучшего среднеквадратичного приближения $Q_0(x)$, $Q_1(x)$, $Q_2(x)$, $Q_3(x)$ и оценить погрешности каждого приближения $\bar{d}(f, Q_0)$, $\bar{d}(f, Q_1)$, $\bar{d}(f, Q_2)$, $\bar{d}(f, Q_3)$.

Решение. Составим ортогональные многочлены Чебышева $g_0(x)$, $g_1(x)$, $g_2(x)$, $g_3(x)$ на множестве точек $\{0, 1, 3, 4\}$. Имеем

$$g_0(x) = 1, \quad g_1(x) = x - a, \quad a = \frac{1}{4} \sum_{i=1}^4 x_i = \frac{1}{4} (0+1+3+4) = 2,$$

$$g_1(x) = x - 2, \quad g_2(x) = (x - a_2)g_1(x) - b_2g_0(x),$$

$$\|g_0\|^2 = 4, \quad \|g_1\|^2 = g_1^2(0) + g_1^2(1) + g_1^2(3) + g_1^2(4) = 10,$$

$$a_2 = \frac{\sum_{i=1}^4 x_i g_1^2(x_i)}{\|g_1\|^2} = \frac{0 \cdot 4 + 1 \cdot 1 + 3 \cdot 1 + 4 \cdot 4}{10} = 2,$$

$$b_2 = \frac{\sum_{i=1}^4 x_i g_0(x_i) g_1(x_i)}{\|g_0\|^2} = \frac{0 \cdot 1 \cdot (-2) + 1 \cdot 1 \cdot (-1) + 3 \cdot 1 \cdot 1 + 4 \cdot 1 \cdot 2}{4} = \frac{5}{2},$$

$$g_2(x) = (x-2)^2 - \frac{5}{2},$$

$$\|g_2\|^2 = \sum_{i=1}^4 g_2^2(x_i) = \frac{9}{4} + \frac{9}{4} + \frac{9}{4} + \frac{9}{4} = 9,$$

$$g_3(x) = (x-a_3)g_2(x) - b_3g_1(x),$$

$$a_3 = \frac{\sum_{i=1}^4 x_i g_2^2(x_i)}{\|g_2\|^2} = \frac{0 \cdot \frac{9}{4} + 1 \cdot \frac{9}{4} + 3 \cdot \frac{9}{4} + 4 \cdot \frac{9}{4}}{9} = 2,$$

$$b_3 = \frac{\sum_{i=1}^4 x_i g_1(x_i) g_2(x_i)}{\|g_1\|^2} = \frac{9}{10},$$

$$g_3(x) = (x-2)^3 - \frac{17}{5}(x-2),$$

$$\|g_3\|^2 = \sum_{i=1}^4 g_3^2(x_i) = \frac{36}{25} + \frac{144}{25} + \frac{144}{25} + \frac{36}{25} = \frac{72}{5}.$$

Значения многочленов $g_0(x)$, $g_1(x)$, $g_2(x)$, $g_3(x)$ на множестве точек $\{0, 1, 3, 4\}$ представлены в таблице

i	x_i	$g_0(x_i)$	$g_1(x_i)$	$g_2(x_i)$	$g_3(x_i)$
1	0	1	-2	1,5	-1,2
2	1	1	-1	-1,5	2,4
3	3	1	1	-1,5	-2,4
4	4	1	2	1,5	1,2

Многочлены наилучшего приближения имеют вид

$$Q_0(x) = C_0 g_0(x),$$

$$Q_1(x) = C_0 g_0(x) + C_1 g_1(x),$$

$$Q_2(x) = C_0 g_0(x) + C_1 g_1(x) + C_2 g_2(x),$$

$$Q_3(x) = C_0 g_0(x) + C_1 g_1(x) + C_2 g_2(x) + C_3 g_3(x).$$

Здесь коэффициенты Фурье, определенные по формулам (27), таковы:

$$C_0 = \frac{7}{4}, \quad C_1 = -\frac{3}{10}, \quad C_2 = \frac{5}{6}, \quad C_3 = -\frac{1}{3}.$$

Квадрат наименьшего среднеквадратичного отклонения определяется соотношением (28) и, в частности, для $\sigma^2(f, Q_3)$ имеем

$$\begin{aligned} \sigma^2(f, Q_3) &= \|f\|^2 - \sum_{k=0}^3 C_k^2 \|g_k\|^2 = 21 - \left[\frac{49}{16} \cdot 4 + \frac{9}{100} \cdot 10 + \frac{25}{36} \cdot 9 + \right. \\ &\quad \left. + \frac{1}{9} \cdot \frac{72}{5} \right] = 21 - \left[\frac{49}{4} + \frac{9}{10} + \frac{25}{4} + \frac{8}{5} \right] = 0. \end{aligned}$$

Тем самым найдены алгебраические многочлены наилучшего среднеквадратичного приближения $Q_0(x)$, $Q_1(x)$, $Q_2(x)$, $Q_3(x)$ и погрешности каждого из приближений $\bar{\sigma}(f, Q_0)$, $\bar{\sigma}(f, Q_1)$, $\bar{\sigma}(f, Q_2)$, $\bar{\sigma}(f, Q_3)$:

$$\begin{aligned} Q_0(x) &= \frac{7}{4}, & \bar{\sigma}(f, Q_0) &= 0,645; \\ Q_1(x) &= -\frac{3}{10}x + \frac{47}{20}, & \bar{\sigma}(f, Q_1) &= 0,611; \\ Q_2(x) &= \frac{5}{6}x^2 - \frac{109}{30}x + \frac{18}{5}, & \bar{\sigma}(f, Q_2) &= 0,276; \\ Q_3(x) &= -\frac{1}{3}x^3 + \frac{17}{6}x^2 - \frac{13}{2}x + 4, & \bar{\sigma}(f, Q_3) &= 0. \end{aligned}$$

Значения алгебраических многочленов наилучшего среднеквадратичного приближения в точках x_i представлены в таблице

i	x_i	$Q_0(x_i)$	$Q_1(x_i)$	$Q_2(x_i)$	$Q_3(x_i)$
1	0	1,75	2,35	3,6	4,00
2	1	1,75	2,05	0,8	0,00
3	3	1,75	1,45	0,2	1,00
4	4	1,75	1,15	2,4	2,00

Отметим, что $Q_3(0) = 4$, $Q_3(1) = 0$, $Q_3(3) = 1$, $Q_3(4) = 2$.

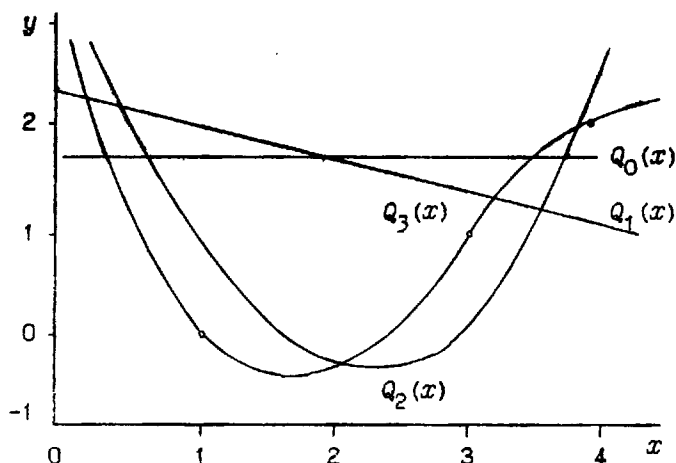


Рис. 21

Графики функций $Q_0(x)$, $Q_1(x)$, $Q_2(x)$, $Q_3(x)$ изображены на рис. 21; на том же рисунке отмечены точки $(1, 0)$, $(3, 1)$, $(4, 2)$. ■

Упражнения

Функцию $f(x)$, полученную, например, в результате эксперимента и определенную таблицей, аппроксимировать алгебраическими многочленами наилучшего среднеквадратичного приближения $Q_2(x)$ и $Q_3(x)$; оценить погрешности $\bar{d}(f, Q_2)$, $\bar{d}(f, Q_3)$. Изобразить графики функций $Q_2(x)$, $Q_3(x)$ и отметить экспериментальные точки в той же системе координат.

1.

x_i	1	2	3	4	5
y_i	1,1	1,4	1,6	1,7	1,9

2.

x_i	1	2	3	4	5
y_i	1,05	1,55	1,7	1,75	1,8

3.

x_i	2	3	4	5	6
y_i	0,4	0,55	0,13	0,09	0,07

4.

x_i	1	2	3	4	5
y_i	7,5	6,2	5,5	3,5	3

5.

x_i	1	2	3	4	5
y_i	8,2	5,9	4,9	4	3,2

6.

x_i	1	2	3	4	5
y_i	7,2	5,9	4,9	4	3,2

7.

x_i	1	2	3	4	5
y_i	7,1	6,1	4,9	4	3,1

8.

x_i	1	2	3	4	5
y_i	0,55	0,7	0,77	0,82	0,85

9.

x_i	1	2	3	4	5
y_i	1,1	1,55	1,9	2,3	2,6

10.

x_i	1	2	3	4	5
y_i	1,1	1,55	1,9	2,25	2,5

11.

x_i	1	2	3	4	5
y_i	5,1	4,4	3,2	2,7	2,55

12.

x_i	1	2	3	4	5
y_i	5,1	3,4	3,2	2,7	2,55

13.

x_i	1	2	3	4	5
y_i	1,9	5,5	10	15	21

14.

x_i	1	2	3	4	5
y_i	3	3,5	3,67	3,75	3,8

15.

x_i	1	2	3	4	5
y_i	0,25	0,09	0,07	0,05	0,04

16.

x_i	1	2	3	4	5
y_i	0,25	0,111	0,071	0,053	0,042

17.

x_i	1	2	3	4	5
y_i	0,20	0,28	0,33	0,36	0,38

18.

x_i	1	2	3	4	5
y_i	4,8	5,76	6,912	8,294	9,95

19.

x_i	1	2	3	4	5
y_i	1	3,08	4,3	5,16	5,83

20.

x_i	1	2	3	4	5
y_i	0,33	0,5	0,6	0,67	0,71

21.

x_i	1	2	3	4	5
y_i	1,5	1,75	1,83	1,87	1,9

22.

x_i	1	2	3	4	5
y_i	1	0,2	0,11	0,077	0,059

23.

x_i	1	2	3	4	5
y_i	1	0,4	0,33	0,31	0,29

24.

x_i	1	2	3	4	5
y_i	2,25	3,37	5,06	7,59	11,4

25.

x_i	1	2	3	4	5
y_i	2	2,69	3,1	3,39	3,61

§5. МЕТОД НАИМЕНЬШИХ КВАДРАТОВ. ЭМПИРИЧЕСКИЕ ФОРМУЛЫ

Пусть данные некоторого эксперимента представлены в виде таблицы значений переменных x и y :

x_t	x_1	x_2		x_m
y_t	y_1	y_2		y_m

Можно поставить задачу об отыскании аналитической зависимости между x и y , т. е. некоторой формулы $y=f(x)$, явным образом выражающей y как функцию x . Естественно требовать, чтобы график искомой функции $y=f(x)$ изменялся плавно и не слишком уклонялся от экспериментальных точек (x_t, y_t) . Поиск такой функциональной зависимости называют "сглаживанием" экспериментальных данных.

Задачу о сглаживании экспериментальных данных можно решать, используя метод наименьших квадратов. Согласно методу наименьших квадратов указывается вид эмпирической формулы

$$y = Q(x, a_0, a_1, \dots, a_n), \quad (30)$$

где $a_0, a_1, \dots, a_n$ — числовые параметры.

Наилучшими значениями параметров $a_0, a_1, \dots, a_n$ (которые обозначим $\tilde{a}_0, \tilde{a}_1, \dots, \tilde{a}_n$) считаются те, для которых сумма квадратов отклонений функции $Q(x, a_0, a_1, \dots, a_n)$ от экспериментальных точек (x_t, y_t) ($t=1, 2, \dots, m$) является минимальной, т.е. функция

$$S(a_0, a_1, \dots, a_n) = \sum_{t=1}^m (Q(x_t, a_0, a_1, \dots, a_n) - y_t)^2 \quad (31)$$

в точке $(\tilde{a}_0, \tilde{a}_1, \dots, \tilde{a}_n)$ достигает минимума. Отсюда, используя необходимые условия экстремума функции нескольких переменных, получаем систему уравнений для определения параметров $\tilde{a}_0, \dots, \tilde{a}_n$:

$$\frac{\partial S}{\partial a_0} = 0, \quad \frac{\partial S}{\partial a_1} = 0, \dots, \quad \frac{\partial S}{\partial a_n} = 0. \quad (32)$$

Если система (32) имеет единственное решение $\tilde{a}_0, \tilde{a}_1, \dots, \tilde{a}_n$, то оно является искомым и аналитическая зависимость между экспе-

риментальными данными определяется формулой

$$y = f(x) = Q(x, \tilde{a}_0, \tilde{a}_1, \dots, \tilde{a}_n).$$

Заметим, что в общем случае система (32) нелинейна.

Рассмотрим подробнее аппроксимирующие зависимости (30) с двумя параметрами: $y = Q(x, \alpha, \beta)$. Используя соотношения (32) и опуская несложные выкладки, получим систему двух уравнений с двумя неизвестными α и β :

$$\begin{cases} \sum_{i=1}^m [Q(x_i, \alpha, \beta) - y_i] \frac{\partial Q(x_i, \alpha, \beta)}{\partial \alpha} = 0, \\ \sum_{i=1}^m [Q(x_i, \alpha, \beta) - y_i] \frac{\partial Q(x_i, \alpha, \beta)}{\partial \beta} = 0. \end{cases} \quad (33)$$

В частном случае аппроксимации экспериментальных данных с помощью линейной функции имеем

$$y = Q(x, k, b) = kx + b, \quad \frac{\partial Q}{\partial k} = x, \quad \frac{\partial Q}{\partial b} = 1.$$

Система (33) для этого случая является линейной относительно неизвестных k и b :

$$\begin{cases} \sum_{i=1}^m [(kx_i + b) - y_i] = 0, \\ \sum_{i=1}^m [(kx_i + b) - y_i] x_i = 0 \end{cases} \iff \begin{cases} bn + k \sum_{i=1}^m x_i = \sum_{i=1}^m y_i, \\ b \sum_{i=1}^m x_i + k \sum_{i=1}^m x_i^2 = \sum_{i=1}^m x_i y_i. \end{cases} \quad (34)$$

Пусть для переменных x и y соответствующие значения экспериментальных данных (x_i, y_i) не располагаются вблизи прямой. Тогда выбирают новые переменные

$$X = \varphi(x, y), \quad Y = \psi(x, y) \quad (35)$$

так, чтобы преобразованные экспериментальные данные

$$X_i = \varphi(x_i, y_i), \quad Y_i = \psi(x_i, y_i) \quad (36)$$

в новой системе координат (X, Y) давали точки (X_i, Y_i) , менее уклоняющиеся от прямой. Для аппроксимирующей прямой

$$Y = kX + b \quad (37)$$

числа k и b можно определить из уравнений (34), где вместо x_i и

y_i подставляют соответствующие значения X_i и Y_i . Нахождение зависимостей (35) называют **выравниванием экспериментальных данных**.

Функциональная зависимость $y=f(x)$ определена неявно уравнением $\Phi(x,y) = k\varphi(x,y) + b$, разрешенным относительно y в частных случаях.

Пример 1. Установить вид эмпирической формулы $y=f(x)$, используя аппроксимирующие зависимости (30) с двумя параметрами α и β , и определить наилучшие значения параметров, если опытные данные представлены таблицей

x_i	1	2	3	4	5
y_i	7,1	27,8	62,1	110	161

Решение. Здесь экспериментальные точки (x_i, y_i) не располагаются вблизи прямой. Положим $X = \ln x$, $Y = \ln y$ и составим таблицу экспериментальных данных в новых переменных X_i и Y_i :

X_i	0,000	0,693	1,099	1,386	1,609
Y_i	1,960	3,325	4,129	4,700	5,081

Точки (X_i, Y_i) лежат приблизительно на прямой (рис.22). Наилучшие значения параметров k и b эмпирической зависимости $Y = kX + b$ (в новых переменных) находятся из системы уравнений (34):

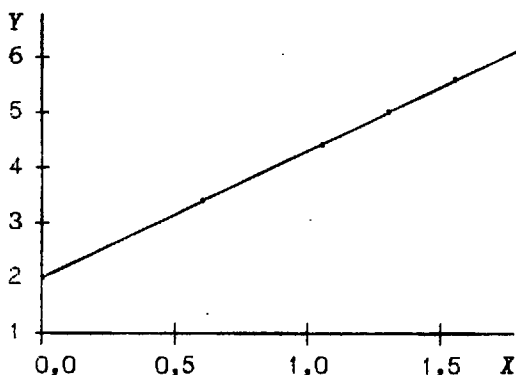


Рис.22

$$\begin{cases} b n + k \sum_{i=1}^m X_i = \sum_{i=1}^m Y_i, \\ b \sum_{i=1}^m X_i + k \sum_{i=1}^m X_i^2 = \sum_{i=1}^m X_i Y_i, \end{cases} \longleftrightarrow \begin{cases} 5 b + 4,787 k = 19,196, \\ 4,787 b + 6,200 k = 21,535. \end{cases}$$

Решив эту систему, получим $b = 1,97$, $k = 1,95$. Неявное уравнение, выражающее связь между переменными x и y , имеет вид

$$\ln y = 1,95 \ln x + 1,97.$$

Легко получить явную зависимость между x и y в виде степенной функции

$$y = e^{1,97} x^{1,95} = 7,16 x^{1,95}. \quad (38)$$

Сравнение экспериментальных данных с результатами вычислений по эмпирической формуле (38) в соответствующих точках представлено в виде таблицы

x_i	1	2	3	4	5
y_i	7,1	27,8	62,1	110	161
$y=7,16 x^{1,95}$	7,16	27,703	61,081	107,04	165,39

Формула (38) является частным случаем аппроксимирующей зависимости с двумя параметрами, имеющей вид

$$Q(x, \alpha, \beta) = \alpha x^\beta.$$

Параметры α и β этой зависимости можно было бы найти из нелинейных уравнений (33) непосредственно. Однако применение способа выравнивания существенно упрощает вычисления параметров. В данном случае $\alpha = e^b$, $\beta = k$. ■

Рекомендации по выравниванию экспериментальных данных и аппроксимирующие зависимости с двумя параметрами приведены в таблице на с. 85.

Одну из шести предложенных формул преобразования к переменным (X, Y) следует выбирать одновременно с проверкой применения линейной зависимости к исходным данным (x_i, y_i) ($i=1, 2, \dots, m$). Условием выбора наилучшей эмпирической формулы является наименьшее отклонение исходных или преобразованных экспериментальных данных

№	Выравнивание данных (преобразование переменных)	Эмпирическая формула
1	$X = x, \quad Y = xy$	$y = \alpha + \frac{\beta}{x}, \quad \alpha=k, \quad \beta=b$
2	$X = x, \quad Y = \frac{1}{y}$	$y = \frac{1}{\alpha x + \beta}, \quad \alpha=k, \quad \beta=b$
3	$X = x, \quad Y = \frac{x}{y}$	$y = \frac{x}{\alpha x + \beta}, \quad \alpha=k, \quad \beta=b$
4	$X = x, \quad Y = \ln y$	$y = \alpha \beta^x, \quad \alpha=e^b, \quad \beta=e^k$
5	$X = \ln x, \quad Y = y$	$y = \alpha \ln x + \beta, \quad \alpha=k, \quad \beta=b$
6	$X = \ln x, \quad Y = \ln y$	$y = \alpha x^\beta, \quad \alpha=e^b, \quad \beta=k$

от прямой. Уклонение данных от прямой в каждом варианте выравнивания данных будем определять величиной

$$d_j = \left[\frac{\sum_{i=1}^m (Y_i - k_j X_i - b_j)^2}{\sum_{i=1}^m Y_i^2} \right]^{\frac{1}{2}}.$$

Для наилучшей эмпирической формулы величина d является наименьшей, т.е. $d = \min_{0 \leq j \leq 6} \{d_j\}$ ($j = 0$ для случая, когда $X_i = x_i$ и $Y_i = y_i$).

Естественно, что если не удастся удовлетворительно построить функциональную зависимость, используя вид эмпирической формулы с двумя параметрами, то можно продолжать поиски среди формул с большим числом параметров.

Пример 2. Опытные данные определены таблицей

x_i	0	1	3	4
y_i	4	0	1	2

Установить вид эмпирической формулы $y=f(x)$, используя аппрокси-

мирующую зависимость с тремя параметрами a , b и c , имеющую вид

$$Q(x, a, b, c) = ax^2 + bx + c.$$

Решение. Здесь соотношение (31) примет вид

$$S(a, b, c) = \sum_{i=1}^m (ax_i^2 + bx_i + c - y_i)^2.$$

Для нахождения a , b , и c составим систему уравнений вида (32):

$\frac{\partial S}{\partial a} = 0$, $\frac{\partial S}{\partial b} = 0$, $\frac{\partial S}{\partial c} = 0$. Отсюда получаем систему трех линейных уравнений с тремя неизвестными:

$$\begin{cases} \sum_{i=1}^m [ax_i^2 + bx_i + c - x_i] x_i^2 = 0, \\ \sum_{i=1}^m [ax_i^2 + bx_i + c - y_i] x_i = 0, \Leftrightarrow \\ \sum_{i=1}^m [ax_i^2 + bx_i + c - y_i] = 0 \end{cases}$$

$$\Rightarrow \begin{cases} a \sum_{i=1}^4 x_i^4 + b \sum_{i=1}^4 x_i^3 + c \sum_{i=1}^4 x_i^2 = \sum_{i=1}^4 y_i x_i^2, \\ a \sum_{i=1}^4 x_i^3 + b \sum_{i=1}^4 x_i^2 + c \sum_{i=1}^4 x_i = \sum_{i=1}^4 y_i x_i, \Leftrightarrow \\ a \sum_{i=1}^4 x_i^2 + b \sum_{i=1}^4 x_i + c \cdot 4 = \sum_{i=1}^4 y_i \end{cases}$$

$$\Rightarrow \begin{cases} a \cdot 338 + b \cdot 92 + c \cdot 26 = 41, \\ a \cdot 92 + b \cdot 26 + c \cdot 8 = 11, \\ a \cdot 26 + b \cdot 8 + c \cdot 4 = 7. \end{cases}$$

Решением этой системы являются числа $a = \frac{5}{6}$, $b = -\frac{109}{30}$, $c = \frac{18}{5}$.

Эмпирическая формула представляет собой функцию

$$y = f(x) = \frac{5}{6} x^2 - \frac{109}{30} x + \frac{18}{5},$$

совпадающую с алгебраическим многочленом наилучшего среднеквадратичного приближения $Q_2(x)$ на множестве точек $\{0, 1, 3, 4\}$ (см. пример 2 §4). ■

Упражнения

Выполнить упражнения 1–25 из §4 в соответствии со следующим заданием.

1. Нанести экспериментальные точки (x_i, y_i) на координатную сетку (x, y) .
2. Выбрать одну из шести предложенных формул преобразования к переменным (X, Y) так, чтобы преобразованные экспериментальные данные (X_i, Y_i) наименее уклонялись от прямой.
3. Методом наименьших квадратов найти наилучшие значения параметров k и b в уравнении прямой (37).
4. Найти явный вид эмпирической формулы $y=Q(x, \alpha, \beta)$ и построить график эмпирической функции.

ИНТЕРПОЛИРОВАНИЕ ФУНКЦИИ

§1. ИНТЕРПОЛЯЦИОННАЯ ФОРМУЛА ЛАГРАНЖА

Пусть функция $y=f(x)$ определена таблицей

x_t	x_0	x_1		x_n
y_t	y_0	y_1		y_n

Значения аргументов $\{x_t\}$ ($t=0,1,\dots,n$) будем называть **узлами интерполяции**.

Задачей интерполяции является построение многочлена $L(x)$, значения которого в узлах интерполяции x_t равны соответствующим значениям заданной функции, т.е.

$$L(x_t) = y_t \quad (t=0,1,\dots,n).$$

Интерполяционной формулой Лагранжа называется формула, представляющая многочлен $L(x)$ в виде

$$L(x) = \sum_{t=0}^n y_t p_t(x), \quad (1)$$

где $p_t(x)$ — многочлен степени n , принимающий значение, равное единице в узле x_t и нулю в остальных узлах x_k ($k \neq t$) ($t, k=0,1,\dots,n$), и имеющий вид

$$p_t(x) = \frac{(x-x_0)(x-x_1)\dots(x-x_{t-1})(x-x_{t+1})\dots(x-x_n)}{(x_t-x_0)(x_t-x_1)\dots(x_t-x_{t-1})(x_t-x_{t+1})\dots(x_t-x_n)}.$$

Многочлен $L(x)$ называют **интерполяционным многочленом Лагранжа**. Заметим, что степень многочлена Лагранжа не превышает числа n .

Пример 1. Пользуясь интерполяционной формулой Лагранжа, составить уравнение прямой, проходящей через точки $P_0(x_0, y_0)$ и $P_1(x_1, y_1)$, если $x_0=-1$, $y_0=-3$, $x_1=2$, $y_1=4$.

Решение. В данном случае многочлен Лагранжа примет вид

$$L(x) = y_0 \frac{x-x_1}{x_0-x_1} + y_1 \frac{x-x_0}{x_1-x_0} = -3 \frac{x-2}{-1-2} + 4 \frac{x+1}{2+1} = \frac{7}{3}x - \frac{2}{3}.$$

Уравнение искомой прямой есть $y = \frac{7}{3}x - \frac{2}{3}$. ■

Пусть функция $y=f(x)$ определена на отрезке $[x_0, x_n]$. Возьмем на этом отрезке множество точек $\{x_i\}$ ($i=0, 1, \dots, n$) и выберем их в качестве узлов интерполяции. Построив многочлен Лагранжа $L(x)$ для системы узлов $\{x_i\}$, положим

$$f(x) \approx L(x) \quad \forall x \in [x_0, x_n].$$

При этом в узлах интерполяции имеем

$$f(x_i) = L(x_i) \quad (i=0, 1, \dots, n).$$

Если функция $f(x)$ на отрезке $[x_0, x_n]$ имеет непрерывные производные до $(n+1)$ -го порядка включительно, то погрешность интерполяционной формулы в каждой точке этого отрезка оценивается неравенством

$$|f(x) - L(x)| \leq \frac{M_{n+1}}{(n+1)!} |\Pi_{n+1}(x)|, \quad (2)$$

где $M_{n+1} = \max_{x_0 \leq x \leq x_n} |f^{(n+1)}(x)|$, $\Pi_{n+1}(x) = (x-x_0)(x-x_1)\dots(x-x_n)$.

Пример 2. Построить многочлен Лагранжа второй степени, аппроксимирующий функцию $y=\sin x$ на отрезке $\left[0, \frac{\pi}{4}\right]$, если заданы значения функции в трех узлах интерполяции:

x	$x_0=0$	$x_1=\pi/6=0,5235988$	$x_2=\pi/4=0,7853982$
$\sin x$	$y_0=0$	$y_1=0,5$	$y_2=0,7071068$

С помощью интерполяционной формулы вычислить приближенное значение $\sin \frac{\pi}{12}$ и оценить погрешность результата вычислений.

Решение. Многочлен Лагранжа для трех узлов интерполяции запишется так:

$$L(x) = y_0 \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1 \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2 \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

или

$$L(x) = 0,5 \frac{x \left[x - \frac{\pi}{4} \right]}{\frac{\pi}{6} \left[\frac{\pi}{6} - \frac{\pi}{4} \right]} + 0,707 \frac{x \left[x - \frac{\pi}{6} \right]}{\frac{\pi}{4} \left[\frac{\pi}{4} - \frac{\pi}{6} \right]} = -2,064 \frac{x^2}{\pi^2} + 3,344 \frac{x}{\pi}.$$

При $x = \frac{\pi}{12} = 0,2617994$ получим $L \left(\frac{\pi}{12} \right) = 0,264298$.

С помощью неравенства (2) находим оценку погрешности. Имеем

$$\left| \sin\left[\frac{\pi}{12}\right] - L\left[\frac{\pi}{12}\right] \right| \leq \frac{M_3}{3!} \left| \Pi_3\left[\frac{\pi}{12}\right] \right|,$$

где $\Pi_3(x) = (x-x_0)(x-x_1)(x-x_2) = x\left(x - \frac{\pi}{6}\right)\left(x - \frac{\pi}{4}\right)$ и

$$\Pi_3\left[\frac{\pi}{12}\right] = \frac{\pi}{12} \left[-\frac{\pi}{12}\right] \left[-\frac{\pi}{6}\right] = 2\left[\frac{\pi}{12}\right]^3.$$

Так как $f(x) = \sin x$, $f'(x) = \cos x$, $f''(x) = -\sin x$, $f'''(x) = -\cos x$, то

$$M_3 = \max_{0 \leq x \leq \frac{\pi}{4}} |f'''(x)| = \max_{0 \leq x \leq \frac{\pi}{4}} |-\cos x| = 1$$

и, следовательно,

$$\left| \sin\left[\frac{\pi}{12}\right] - L\left[\frac{\pi}{12}\right] \right| \leq \frac{1}{3} \left[\frac{\pi}{12}\right]^3 \approx 0,006.$$

Итак, $\sin \frac{\pi}{12} \approx 0,264 \pm 0,006$. Заметим, что это значение с шестью

верными цифрами есть $\sin \frac{\pi}{12} = 0,258819$. ■

Упражнения

1. Функция $f(x)$ аппроксимируется интерполяционным многочленом Лагранжа второй степени для системы трех равномерно расположенных на отрезке узлов. Оценить погрешность интерполяции в точке ξ :

а) $f(x) = \cos x^2$, $\xi = \frac{\pi}{12}$ $\left\{ 0 \leq x \leq \frac{\pi}{4} \right\}$;

б) $f(x) = x \cos x$, $\xi = \frac{\pi}{12}$ $\left\{ 0 \leq x \leq \frac{\pi}{4} \right\}$;

в) $f(x) = e^{-x^2}$, $\xi = 0,3$ $(0 \leq x \leq 0,4)$.

2. Функция $f(x)$ определена на отрезке $[1,00; 1,20]$ (см. таблицу на с. 91). Найти значения многочлена Лагранжа, интерполирующего функцию $f(x)$ на этом отрезке по системе трех равномерно расположенных узлов (с шагом 0,1), в точках 1,05; 1,09; 1,13; 1,15; 1,17. Полученные результаты сравнить с табличными значениями и дать оценку точности интерполяции, используя неравенство (2).

x	e^x	e^{-x}	$\operatorname{sh} x$	$\operatorname{ch} x$	$\sin x$	$\cos x$	$\ln x$
1,00	2,7183	0,3679	1,1752	1,5431	0,8415	0,5403	0,0000
01	2,7456	0,3642	1,1907	1,5549	0,8468	0,5319	0,0100
02	2,7732	0,3606	1,2063	1,5669	0,8521	0,5234	0,0198
03	2,8011	0,3570	1,2220	1,5790	0,8573	0,5148	0,0296
04	2,8292	0,3535	1,2379	1,5913	0,8624	0,5062	0,0392
05	2,8577	0,3499	1,2539	1,6038	0,8674	0,4976	0,0488
06	2,8864	0,3465	1,2700	1,6164	0,8724	0,4889	0,0583
07	2,9154	0,3430	1,2862	1,6292	0,8772	0,4801	0,0677
08	2,9447	0,3396	1,3025	1,6421	0,8820	0,4713	0,0770
09	2,9743	0,3362	1,3190	1,6552	0,8866	0,4625	0,0862
1,10	3,0042	0,3329	1,3356	1,6685	0,8912	0,4536	0,0953
11	3,0344	0,3296	1,3524	1,6820	0,8957	0,4447	0,1044
12	3,0649	0,3263	1,3693	1,6956	0,9001	0,4357	0,1133
13	3,0957	0,3230	1,3863	1,7093	0,9044	0,4267	0,1222
14	3,1268	0,3198	1,4035	1,7233	0,9086	0,4176	0,1310
15	3,1582	0,3166	1,4208	1,7374	0,9128	0,4085	0,1398
16	3,1899	0,3135	1,4382	1,7517	0,9168	0,3993	0,1484
17	3,2220	0,3104	1,4558	1,7662	0,9208	0,3902	0,1570
18	3,2544	0,3073	1,4735	1,7808	0,9246	0,3809	0,1655
19	3,2871	0,3042	1,4914	1,7957	0,9284	0,3717	0,1740
1,20	3,3201	0,3012	1,5095	1,8107	0,9320	0,3624	0,1823

Замечание. В процессе решения задачи 1 могут возникнуть трудности при оценке величины $M_3 = \max_{0 \leq x \leq \frac{\pi}{4}} |f'''(x)|$.

Проведем эту оценку в задаче 1а, где $f(x) = \cos x^2$.

Строгое вычисление величины $\max_{0 \leq x \leq \frac{\pi}{4}} |f'''(x)|$ требует нахождения

точек экстремума функции $f'''(x)$ на отрезке $\left[0, \frac{\pi}{4}\right]$,

вычисления значений функции в точках экстремума и на концах отрезка и, наконец, выбор необходимого значения M_3 .

Для этого вычислим производные функции $f(x)$ вплоть до четвертого порядка:

$$f'(x) = -2x \sin x^2, \quad f''(x) = -2(\sin x^2 + 2x^2 \cos x^2),$$

$$f'''(x) = -4x(3\cos x^2 - 2x^2 \sin x^2),$$

$$f^{(4)}(x) = -4(3\cos x^2 - 4x^4 \cos x^2 - 12x^2 \sin x^2).$$

Точки экстремума функции $f'''(x)$ являются корнями уравнения

$$f^{(4)}(x) = 0, \quad 4(3\cos x^2 - 4x^4 \cos x^2 - 12x^2 \sin x^2) = 0, \quad x \in \left[0, \frac{\pi}{4}\right].$$

Эти корни находятся с помощью численных методов, изложенных в гл. 3.

Завышенную, но вполне удовлетворительную оценку величины M_3 получим с помощью следующих более простых преобразований.

На отрезке $\left[0, \frac{\pi}{4}\right]$ справедливы неравенства

$$x \cos x^2 \geq 0, \quad x^3 \sin x^2 \geq 0, \quad 3x \cos x^2 \geq 2x^3 \sin x^2.$$

Поэтому

$$M_3 = \max_{0 \leq x \leq \frac{\pi}{4}} |f'''(x)| = 4 \max_{0 \leq x \leq \frac{\pi}{4}} (3x \cos x^2 - 2x^3 \sin x^2) \leq$$

$$\leq 4 \left[\max_{0 \leq x \leq \frac{\pi}{4}} 3x \cos x^2 - \min_{0 \leq x \leq \frac{\pi}{4}} 2x^3 \sin x^2 \right] = 12 \max_{0 \leq x \leq \frac{\pi}{4}} x \cos x^2 < 3\pi.$$

Итак, получена несколько завышенная оценка $M_3 < 3\pi$.

§2. ИНТЕРПОЛИРОВАНИЕ ФУНКЦИЙ КУБИЧЕСКИМИ СПЛАЙНАМИ

Пусть отрезок $[a, b]$ разбит на n частей точками $\{x_i\}$:
 $a=x_0 < x_1 < x_2 < \dots < x_{i-1} < x_i < \dots < x_n=b$.

Сплайном k -й степени называется функция, представляющая собой многочлен не выше k -й степени на каждом из последовательно примыкающих друг к другу интервалов (x_{i-1}, x_i) ($i=1, 2, \dots, n$), причем в точках стыка двух интервалов x_i ($i=1, \dots, n-1$) функция непрерывна вместе со своими производными до порядка не выше k .

Например, непрерывная кусочно-линейная функция (ломаная) является сплайном первой степени с производной, терпящей разрыв в точках излома.

Пусть на отрезке $[a, b]$ определена функция $y=f(x)$, значения которой в точках x_i равны $y_i=f(x_i)$.

Задача интерполяции функции $y=f(x)$ на отрезке $[a, b]$ кубическим сплайном (сплайном третьей степени) состоит в нахождении функции $S(x)$, равной многочлену третьей степени $S_i(x)$ на каждом отрезке $[x_{i-1}, x_i]$ ($i=1, 2, \dots, n$), т.е.

$$S(x) = S_i(x) = a_0^i x^3 + a_1^i x^2 + a_2^i x + a_3^i, \quad x \in [x_{i-1}, x_i], \quad (3)$$

причем значения сплайна в узлах интерполяции x_i равны соответствующим значениям заданной функции y_i и сплайн-функция непрерывна в узлах интерполяции вместе с производными первого и второго порядков:

$$S(x_i) = S_{i+1}(x_i) = y_i \quad (i=0, 1, 2, \dots, n-1), \quad S(x_n) = S_n(x_n) = y_n, \quad (4)$$

$$S_i(x_i) = S_{i+1}(x_i) \quad (i=1, 2, \dots, n-1), \quad (5)$$

$$S'_i(x_i) = S'_{i+1}(x_i) \quad (i=1, 2, \dots, n-1), \quad (6)$$

$$S''_i(x_i) = S''_{i+1}(x_i) \quad (i=1, 2, \dots, n-1). \quad (7)$$

Условия (4)–(7) дают $4n-2$ линейных алгебраических уравнений для определения $4n$ неизвестных коэффициентов a_p^i ($p=0, 1, 2, 3$; $i=1, 2, \dots, n$) при соответствующих степенях x в многочленах $S_i(x)$.

Можно показать, что интерполяционный кубический сплайн для

функции $y=f(x)$ существует и является единственным, если вместе с уравнениями (4)–(7) удовлетворяется какая-либо пара дополнительных условий (краевых условий) следующего типа:

$$\text{I.} \quad S'(a) = f'(a), \quad S'(b) = f'(b);$$

$$\text{II.} \quad S''(a) = f''(a), \quad S''(b) = f''(b);$$

$$\text{III.} \quad S'(a) = S'(b), \quad S''(a) = S''(b).$$

Рассмотрим случай разбиения отрезка $[a, b]$ на n равных частей с шагом h , для которого $x_0=a$, $x_1=x_0+h, \dots, x_{i+1}=x_i+h, \dots, x_n=b$ и $h=(b-a)/n$. Разберем построение интерполяционного кубического сплайна отдельно для условий I и II типов.

При построении сплайна, удовлетворяющего краевым условиям I типа, введем величины $m_i=S'(x_i)$, называемые иногда наклонами сплайна в точках (узлах) x_i ($i=0, 1, \dots, n$).

Интерполяционный кубический сплайн вида

$$\begin{aligned} S(x) = S_i(x) = & y_{i-1} \frac{(x - x_i)^2 (2(x - x_{i-1}) + h)}{h^3} + \\ & + y_i \frac{(x - x_{i-1})^2 (2(x_i - x) + h)}{h^3} + \\ & + m_{i-1} \frac{(x - x_i)^2 (x - x_{i-1})}{h^2} + m_i \frac{(x - x_{i-1})^2 (x - x_i)}{h^2}, \quad (8) \\ & x \in [x_{i-1}, x_i] \quad (i=1, 2, \dots, n) \end{aligned}$$

удовлетворяет условиям (4), (5), (6) для любых m_i . Из условий (7) и краевых условий I типа можно определить $n+1$ параметр m_i .

Действительно, легко проверить, что $S(x_{i-1})=S_i(x_{i-1})=y_{i-1}$, $S(x_i)=S_i(x_i)=y_i$ ($i=1, 2, \dots, n$). Кроме того, вычисления показывают, что

$$S'(x_i) = S'_i(x_i) = m_i,$$

$$S'(x_i) = S'_{i+1}(x_i) = m_i \quad (i=1, 2, \dots, n-1).$$

Если учесть, что

$$S''_i(x_i) = \frac{2m_{i-1}}{h} + \frac{4m_i}{h} - 6 \frac{y_i - y_{i-1}}{h^2} \quad (i=1, 2, \dots, n-1),$$

$$S''_{i+1}(x_i) = -\frac{4m_i}{h} - \frac{2m_{i+1}}{h} + 6 \frac{y_{i+1} - y_i}{h^2} \quad (i=1, 2, \dots, n-1),$$

а также краевые условия I типа и условия (?), то получим систему из $n+1$ линейных уравнений относительно неизвестных m_i :

$$\begin{cases} m_0 = b_0 = f'(a), \\ m_{i-1} + 4m_i + m_{i+1} = b_i = \frac{3(y_{i+1} - y_{i-1})}{h} \quad (i=1, 2, \dots, n-1), \\ m_n = b_n = f'(b). \end{cases} \quad (9)$$

Решение этой системы позволяет найти значения неизвестных m_i и определить интерполяционный сплайн в виде соотношений (8).

Матрица A системы (9) имеет порядок $n+1$ и является трехдиагональной:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 1 & 4 & 1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 1 & 4 & 1 & \dots & 0 & 0 & \dots & 0 \\ \cdot & \cdot & \cdot & \cdot & \dots & \cdot & \cdot & \dots & \cdot \\ 0 & 0 & 0 & 0 & \dots & 1 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 1 \end{bmatrix}$$

Метод Гаусса (метод исключения неизвестных) для системы (9) значительно упрощается и носит название метода прогонки. Прямой прогонкой находят так называемые прогоночные коэффициенты:

$$L_0=0, \quad M_0=b_0, \quad L_i = \frac{-1}{L_{i-1} + 4}, \quad M_i = L_i(M_{i-1} - b_i) \quad (i=1, 2, \dots, n-1).$$

Обратной прогонкой последовательно определяют неизвестные m_i :

$$\begin{cases} m_n = b_n, \\ m_i = L_i m_{i+1} + M_i \quad (i=n-1, n-2, \dots, 0). \end{cases}$$

Пример 1. На отрезке $\left[0, \frac{\pi}{2}\right]$ построить кубический сплайн с шагом $h = \frac{\pi}{2}$, удовлетворяющий на концах отрезка крайним условиям I типа и интерполирующий функцию $y = \sin x$. С помощью интерполяционной формулы вычислить приближенное значение $\sin \frac{\pi}{6}$ и сравнить его с точным.

Решение. Будем искать кубическую параболу $y=S(x)$, удовлетворя-

ную следующим условиям на концах отрезка $x_0 = 0$ и $x_1 = h = \frac{\pi}{2}$:

$$y_0 = S(x_0) = \sin x_0 = 0,$$

$$y_1 = S(h) = \sin h = 1,$$

$$m_0 = S'(x_0) = \sin' x_0 = \cos x_0 = 1,$$

$$m_1 = S'(h) = \sin' h = \cos h = 0.$$

Подставим значения h , y_0 , y_1 , m_0 , m_1 в формулу (8) и получим сплайн вида

$$S(x) = S_1(x) = \frac{x^2(2(h-x)+h)}{h^3} + \frac{(x-h)^2 x}{h^2}, \quad x \in [0, h];$$

$$S(x) = x - \frac{4(\pi-3)^2}{\pi^2} x^2 - \frac{4(4-\pi)}{\pi^3} x^3 = x - 0,057385x^2 - 0,11074x^3.$$

Тогда $\sin \frac{\pi}{6} \approx S\left[\frac{\pi}{6}\right] = 0,49196983$ (точное значение равно 0,5). ■

Пример 2. На отрезке $\left[0, \frac{\pi}{2}\right]$ построить кубический сплайн с шагом $h = \frac{\pi}{4}$, интерполирующий функцию $y = \sin x$, если заданы значения функции в трех узлах интерполяции:

x	$x_0=0$	$x_1=\pi/4=0,7853982$	$x_2=\pi/2=1,570796$
$\sin x$	$y_0=0$	$y_1=0,7071068$	$y_2=1$

С помощью интерполяционной формулы вычислить приближенное значение $\sin \frac{\pi}{6}$ $\left[\frac{\pi}{6} = 0,5235988\right]$ и сравнить с точным значением 0,5.

Решение. Представим сплайн в виде (8):

$$S(x) = \begin{cases} S_1(x), & 0 \leq x \leq \pi/4; \\ S_2(x), & \pi/4 \leq x \leq \pi/2. \end{cases}$$

При таком представлении должны удовлетворяться уравнения (9):

$$\begin{cases} m_0 = \sin' x_0 = \cos x_0 = 1, \\ m_0 + 4m_1 + m_2 = \frac{3(y_2 - y_0)}{h} = \frac{3}{h}, \\ m_2 = \sin' x_2 = \cos x_2 = 0. \end{cases}$$

Тогда, учитывая, что $1 + 4m_1 = \frac{12}{\pi}$ или $m_1 = \frac{12-\pi}{4\pi} = 0,70493$,

$$S_1(x) = y_1 \frac{x^2(2(h-x)+h)}{h^3} + \frac{(x-h)^2x}{h^2} + m_1 \frac{x^2(x-h)}{h^2}, \quad 0 \leq x \leq h;$$

$$S_2(x) = y_1 \frac{(x-2h)^2(2(x-h)+h)}{h^3} + \frac{(x-h)^2(2(2h-x)+h)}{h^3} + \\ + m_1 \frac{(x-2h)^2(x-h)}{h^2}, \quad h \leq x \leq 2h,$$

получим, в частности, выражение для функции $S_1(x)$:

$$S_1(x) = y_1 \frac{x^2(2(h-x)+h)}{h^3} + \frac{(x-h)^2x}{h^2} + m_1 \frac{x^2(x-h)}{h^2} = \\ = x - 0,0050683975 x^2 - 0,15514782 x^3, \quad 0 \leq x \leq \frac{\pi}{4}.$$

Значение $\sin \frac{\pi}{6} \approx S_1\left[\frac{\pi}{6}\right] = 0,499938$. ■

При построении сплайна, удовлетворяющего краевым условиям II типа, введем величину $\tilde{m}_1 = S''(x_1)$ — значение второй производной сплайна в узле x_i ($i=0,1,\dots,n$).

Уравнения (4), (5), (7) будут удовлетворены, если интерполяционный кубический сплайн представить в виде

$$S(x) = S_i(x) = y_{i-1} \frac{(x_1-x)}{h} + y_i \frac{(x-x_{i-1})}{h} + \\ + \tilde{m}_{i-1} \frac{(x_1-x)^3 - h^2(x_1-x)}{6h} + \tilde{m}_i \frac{(x-x_{i-1})^3 - h^2(x-x_{i-1})}{6h}, \quad (10) \\ x \in [x_{i-1}, x_i] \quad (i=1,2,\dots,n).$$

Учитывая, что

$$S'_1(x_1) = \frac{y_1 - y_{1-1}}{h} + \frac{h\tilde{m}_1 + 2h\tilde{m}_1}{6} \quad (i=1,2,\dots,n-1),$$

$$S'_{i+1}(x_i) = \frac{y_{i+1} - y_i}{h} - \frac{2h\tilde{m}_i + h\tilde{m}_{i+1}}{6} \quad (i=1,2,\dots,n-1),$$

и используя краевые условия II типа и условия (6), получим систему из $n+1$ линейных уравнений относительно неизвестных $\tilde{m}_i$:

$$\begin{cases} \tilde{m}_0 = f''(a), \\ \tilde{m}_{i-1} + 4\tilde{m}_i + \tilde{m}_{i+1} = \frac{6(y_{i+1} - 2y_i + y_{i-1}))}{h^2} \quad (i=1, 2, \dots, n-1), \\ \tilde{m}_n = f''(b). \end{cases} \quad (11)$$

Системы (9) и (11) являются частными случаями системы линейных алгебраических уравнений следующего вида:

$$\begin{cases} u_0 = b_0, \\ u_{i-1} + 4u_i + u_{i+1} = b_i \quad (i=1, 2, \dots, n-1), \\ u_n = b_n. \end{cases} \quad (12)$$

Для функции $f(x)$, имеющей на отрезке $[a, b]$ непрерывные производные до третьего порядка включительно, точность интерполяции ее кубическим сплайном $S(x)$ по точкам равномерного разбиения отрезка с шагом h при любых указанных ранее краевых условиях оценивается следующим неравенством для любых x на отрезке $[a, b]$:

$$|f(x) - S(x)| \leq \frac{5}{2} M_3 h^3, \quad \text{где } M_3 = \max_{a \leq x \leq b} |f'''(x)|. \quad (13)$$

Неравенство (13) дает завышенную оценку точности приближения функции сплайном в точке.

Упражнения

Построить кубический сплайн, интерполирующий функцию $y = f(x)$ на отрезке $[1,00; 1,20]$ для равномерного разбиения с шагом $h=0,04$ при краевых условиях I или II типа.

Функция $y = f(x)$ определена аналитически: 1) e^x ; 2) $\sinh x$; 3) $\cosh x$; 4) $\sin x$; 5) $\cos x$; 6) $\ln x$; 7) e^{-x} .

Найти значение сплайна в точках 1,05; 1,09; 1,13; 1,15; 1,17. Получить оценку точности с помощью неравенства (13) и сравнить значения сплайна в указанных точках с точными значениями функции, приведенными в таблице на с. 91.

ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ

§1. ВЫЧИСЛЕНИЕ ПРОИЗВОДНОЙ ПО ЕЕ ОПРЕДЕЛЕНИЮ

Пусть функция $y=f(x)$ определена в некоторой окрестности точки x_0 и имеет производную в этой точке, т.е. существует предел отношения приращения функции Δy к приращению аргумента Δx при стремлении Δx к нулю:

$$y'(x_0) = f'(x_0) = \lim_{\Delta x \rightarrow 0} \frac{\Delta y}{\Delta x}, \quad \Delta x = x - x_0, \quad \Delta y = f(x_0 + \Delta x) - f(x_0). \quad (1)$$

Значение производной в точке x_0 можно получить, переходя к пределу в (1) по последовательности целых чисел n и полагая, например,

$$\Delta x = (\Delta x)_n = \frac{(\Delta x)_0}{\alpha^n}. \quad \text{Здесь } (\Delta x)_0 - \text{некоторое начальное}$$

приращения аргумента, α - некоторое число, большее единицы, $n = \{0, 1, 2, 3, \dots\}$. Тогда значение производной функции $f(x)$ в точке x_0 запишется так:

$$y'(x_0) = f'(x_0) = \lim_{n \rightarrow \infty} \frac{(\Delta y)_n}{(\Delta x)_n}, \quad (\Delta y)_n = f(x_0 + (\Delta x)_n) - f(x_0).$$

Отсюда получаем приближенное равенство

$$y'(x_0) \approx \frac{(\Delta y)_n}{(\Delta x)_n}. \quad (2)$$

Для функции $y=f(x)$, имеющей непрерывную производную до второго порядка включительно в окрестности точки x_0 , точность приближения производной соотношением (2) можно установить, воспользовавшись формулой Тейлора

$$f(x) = f(x_0) + f'(x_0)\Delta x + \frac{1}{2}f''(\xi)\Delta x^2, \quad \xi \in (x_0, x).$$

Тогда

$$\left| f'(x_0) - \frac{f(x) - f(x_0)}{\Delta x} \right| \leq \frac{M}{2} \Delta x, \quad M = \max_{x_0 \leq \xi \leq x} |f''(\xi)|$$

и окончательно имеем

$$\left| y'(x_0) - \frac{(\Delta y)_n}{(\Delta x)_n} \right| \leq \frac{L}{2} (\Delta x)_0 \alpha^{-n}, \quad L = \max_{x_0 \leq \xi \leq x} |f''(\xi)|.$$

Для достижения заданной точности ε приближения производной при определенном числе вычислений можно использовать неравенство

$$\left| \frac{(\Delta y)_n}{(\Delta x)_n} - \frac{(\Delta y)_{n-1}}{(\Delta x)_{n-1}} \right| < \varepsilon. \quad (3)$$

Пример. Вычислить производную функции $y = \sin x$ в точке $x_0 = \pi/3$ с точностью $\varepsilon = 10^{-3}$ ($\pi/3 = 1,047198$).

Решение. Положим $(\Delta x)_0 = 0,1$; $\alpha = 10$; $(\Delta x)_n = 0,1 \cdot 10^{-n}$, откуда

$$(\Delta y)_n = \sin\left[\frac{\pi}{3} + 0,1 \cdot 10^{-n}\right] - \sin \frac{\pi}{3}.$$

Определим приближенное значение производной:

$$y' \left(\frac{\pi}{3} \right) \approx \frac{(\Delta y)_n}{(\Delta x)_n} = \frac{\sin\left[\frac{\pi}{3} + 0,1 \cdot 10^{-n}\right] - \sin \frac{\pi}{3}}{0,1 \cdot 10^{-n}} \quad (n=0,1,2,\dots).$$

Найдем отношения, аппроксимирующие производную:

$$\frac{(\Delta y)_0}{(\Delta x)_0} = 0,45590189; \quad \frac{(\Delta y)_1}{(\Delta x)_1} = 0,49566158; \quad \frac{(\Delta y)_2}{(\Delta x)_2} = 0,49956690;$$

$$\frac{(\Delta y)_3}{(\Delta x)_3} = 0,49995670; \quad \left| \frac{(\Delta y)_3}{(\Delta x)_3} - \frac{(\Delta y)_2}{(\Delta x)_2} \right| = 0,00038979390 < \varepsilon.$$

Итак, начиная с третьего приближения, в соответствии с оценкой (3) получаем искомое приближение производной данной функции $y'(\pi/3) = \cos(\pi/3) = 0,5$ с точностью, не меньшей чем заданная $\varepsilon = 0,001$. ■

§2. КОНЕЧНО-РАЗНОСТНЫЕ АППРОКСИМАЦИИ ПРОИЗВОДНЫХ

Пусть отрезок $[a, b]$ разбит на n ($n \geq 2$) равных частей точками $\{x_i\}$: $a = x_0 < x_1 < x_2 < \dots < x_{i-1} < x_i < x_{i+1} < \dots < x_n = b$. Разность между соседними значениями аргумента постоянна, т.е. шаг $h = x_i - x_{i-1}$ ($i=1, 2, \dots, n$). Далее, пусть на отрезке $[a, b]$ определена функция $y=f(x)$, значения которой в точках x_i равны $y_i = f(x_i)$ ($i=0, 1, 2, \dots, n$).

Запишем выражения для первой производной функции в точке x_i с помощью отношения конечных разностей:

а) аппроксимация с помощью разностей вперед (правых разностей)

$$y'(x_i) \approx \frac{\Delta y_i}{\Delta x_i}, \quad \Delta x_i = x_{i+1} - x_i = h, \quad \Delta y_i = y_{i+1} - y_i,$$

$$y'(x_i) \approx \frac{y_{i+1} - y_i}{h} \quad (i=0, 1, \dots, n-1); \quad (4)$$

б) аппроксимация с помощью разностей назад (левых разностей)

$$y'(x_i) \approx \frac{\Delta y_i}{\Delta x_i}, \quad \Delta x_i = x_{i-1} - x_i = -h, \quad \Delta y_i = y_{i-1} - y_i,$$

$$y'(x_i) \approx \frac{y_i - y_{i-1}}{h} \quad (i=1, 2, \dots, n); \quad (5)$$

в) аппроксимация с помощью центральных разностей

(точка x_i является центром системы точек x_{i-1} , x_i , x_{i+1})

$$y'(x_i) \approx \frac{\Delta y_i}{\Delta x_i}, \quad \Delta x_i = x_{i+1} - x_{i-1} = 2h, \quad \Delta y_i = y_{i+1} - y_{i-1},$$

$$y'(x_i) \approx \frac{y_{i+1} - y_{i-1}}{2h} \quad (i=1, \dots, n-1). \quad (6)$$

Аппроксимация производной с помощью центральных разностей представляет собой среднее арифметическое соотношений (4) и (5) в точках $\{x_i\}$ ($i=1, \dots, n-1$).

Отметим, что соотношения (4) и (6) не позволяют вычислить производную в точке $x_n = b$, а (5) и (6) — в точке $x_0 = a$.

Можно показать, что для функции $y=f(x)$, имеющей непрерывную производную до второго порядка включительно, погрешность аппроксимации производных разностями вперед и назад имеет один и тот же порядок $O(h)$, а погрешность аппроксимации центральными разностями (6) для функции $y=f(x)$, имеющей непрерывную производную до третьего порядка включительно, имеет порядок $O(h^2)$.

Приближенное значение производной второго порядка в точке x_i выразим через значения функции y_{i-1} , y_i , y_{i+1} . Для этого представим вторую производную с помощью правой разности:

$$y''(x_i) \approx \frac{\Delta y'_i}{\Delta x_i}, \quad \Delta x_i = x_{i+1} - x_i = h, \quad \Delta y'_i = y'_{i+1} - y'_i,$$

а производные первого порядка y'_{i+1} и y'_i — с помощью левых разностей:

$$y'_{i+1} = y'(x_{i+1}) \approx \frac{y_{i+1} - y_i}{h}, \quad y'_i = y'(x_i) \approx \frac{y_i - y_{i-1}}{h}$$

и окончательно получим

$$y''(x_i) \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \quad (i=1, 2, \dots, n-1). \quad (7)$$

Погрешность последней аппроксимации имеет порядок $O(h^2)$ для функции $y=f(x)$, имеющей непрерывную производную до четвертого порядка включительно на отрезке $[a, b]$. Естественно, что представление (7) с помощью конечных разностей позволяет вычислять значения второй производной только во внутренних точках отрезка.

§3. ИСПОЛЬЗОВАНИЕ ИНТЕРПОЛЯЦИОННЫХ МНОГОЧЛЕНОВ ЛАГРАНЖА ДЛЯ ФОРМУЛ ЧИСЛЕННОГО ДИФФЕРЕНЦИРОВАНИЯ

Пусть функция $y=f(x)$ определена на отрезке $[a, b]$ и в точках $\{x_i\}$ ($i=0, 1, 2, \dots, n$) этого отрезка принимает значения $y_i=f(x_i)$.

Разность между соседними значениями аргумента x_i постоянна и является шагом $h=x_i-x_{i-1}$ ($i=1, \dots, n$) разбиения отрезка на n частей, причем $a=x_0$ и $b=x_n$.

Найдем аппроксимации производных первого и второго порядков с помощью значений функций y_i в узловых точках x_i с погрешностью одного и того же порядка в зависимости от шага h , причем этот порядок не ниже, чем достигаемый при конечно-разностной аппроксимации производных для того же шага.

Для того чтобы выразить значения производных через значения функции y_i в узлах интерполяции x_i , построим интерполяционный многочлен Лагранжа $L_m(x)$ степени m , удовлетворяющий условиям

$$L_m(x_k) = f(x_k) = y_k \quad (k=i, i+1, \dots, i+m), \quad i+m \leq n.$$

Многочлен $L_m(x)$ интерполирует функцию $f(x)$ на отрезке $[x_i, x_{i+m}]$.

Дифференцируя многочлен $L_m(x)$, получаем значения производных в точках $\{x_k\}$ ($k=i, i+1, \dots, i+m$).

Если $m=1$, то $L_1(x)$ — линейная функция, график которой проходит через точки (x_i, y_i) и (x_{i+1}, y_{i+1}) . Тогда

$$L_1(x) = -y_i \frac{x-x_{i+1}}{h} + y_{i+1} \frac{x-x_i}{h},$$

$$y'_i = f'(x_i) \approx L'_1(x) = \frac{y_{i+1}-y_i}{h}, \quad y'_{i+1} = f'(x_{i+1}) \approx L'_1(x).$$

Если $m = 2$, то график интерполяционного многочлена Лагранжа $L_2(x)$ — парабола, проходящая через три точки (x_i, y_i) , (x_{i+1}, y_{i+1}) и (x_{i+2}, y_{i+2}) . Вычислим первую и вторую производные многочлена $L_2(x)$ на отрезке $[x_i, x_{i+2}]$:

$$L_2(x) = \frac{1}{2h^2} [y_i(x-x_{i+1})(x-x_{i+2}) - 2y_{i+1}(x-x_i)(x-x_{i+2}) + y_{i+2}(x-x_i)(x-x_{i+1})],$$

$$L'_2(x) = \frac{1}{2h^2} [y_i(2x-x_{i+1}-x_{i+2}) - 2y_{i+1}(2x-x_i-x_{i+2}) + y_{i+2}(2x-x_i-x_{i+1})],$$

$$L''_2(x) = \frac{1}{h^2} (y_i - 2y_{i+1} + y_{i+2}).$$

Первая и вторая производные многочлена Лагранжа $L_2(x)$ в точках x_i, x_{i+1}, x_{i+2} являются приближениями соответствующих производных функции $f(x)$ в этих точках:

$$\left. \begin{aligned} y'_i = f'(x_i) \approx L'_2(x_i) &= \frac{1}{2h} (-3y_i + 4y_{i+1} - y_{i+2}), \\ y'_{i+1} = f'(x_{i+1}) \approx L'_2(x_{i+1}) &= \frac{1}{2h} (-y_i + y_{i+2}), \\ y'_{i+2} = f'(x_{i+2}) \approx L'_2(x_{i+2}) &= \frac{1}{2h} (y_i - 4y_{i+1} + 3y_{i+2}), \end{aligned} \right\} \quad (8)$$

$$y''_i = y''_{i+1} = y''_{i+2} \approx L''_2(x) = \frac{1}{h^2} (y_i - 2y_{i+1} + y_{i+2}). \quad (9)$$

Если функция $f(x)$ на отрезке $[x_i, x_{i+2}]$ имеет непрерывную производную до третьего порядка включительно, то справедливо представление функции в виде суммы:

$$f(x) = L_2(x) + R_2(x), \quad (10)$$

где $R_2(x)$ — остаточный член интерполяционной формулы, причем

$$R_2(x) = \frac{f''(\xi)}{3!} (x-x_i)(x-x_{i+1})(x-x_{i+2}), \quad \xi \in (x_i, x_{i+2}).$$

В этом случае можно дать оценку погрешности приближений производных соотношениями (8) и (9). Дифференцируя (10), получим

$$f'(x) = L'_2(x) + R'_2(x), \quad (11)$$

$$f''(x) = L''_2(x) + R''_2(x). \quad (12)$$

Здесь

$$R'_2(x) = \frac{f''(\xi)}{3!} [(x-x_{i+1})(x-x_{i+2}) + (x-x_i)(x-x_{i+2}) + (x-x_i)(x-x_{i+1})], \quad \xi \in (x_i, x_{i+2}), \quad (13)$$

$$R''_2(x) = \frac{f'''(\xi)}{3} [(x-x_i) + (x-x_{i+1}) + (x-x_{i+2})], \quad \xi \in (x_i, x_{i+2}). \quad (14)$$

Погрешности при вычислении производных в точках x_i , x_{i+1} , x_{i+2} определяются из формул (11)–(14) следующими значениями остатков:

$$R'_2(x_i) = -2R'_2(x_{i+1}) = R'_2(x_{i+2}) = \frac{1}{3} h^2 f'''(\xi), \quad (15)$$

$$R''_2(x_i) = -hf'''(\xi), \quad R''_2(x_{i+1}) = 0, \quad R''_2(x_{i+2}) = hf'''(\xi). \quad (16)$$

Таким образом, равенства (15) показывают, что погрешности аппроксимации первой производной $f'(x)$ с помощью формул (8) имеют один и тот же порядок $O(h^2)$, и естественна следующая рекомендация по их применению на отрезке $[a, b]$ в точках $\{x_i\}$ ($i=0, 1, 2, \dots, n$) при $n \geq 2$:

$$\begin{aligned} y'_0 &\approx \frac{1}{2h} (-3y_0 + 4y_1 - y_2), \\ y'_i &\approx \frac{1}{2h} (-y_{i-1} + y_{i+1}) \quad (i=1, 2, \dots, n-1), \\ y'_n &\approx \frac{1}{2h} (y_{n-2} - 4y_{n-1} + 3y_n). \end{aligned} \quad (17)$$

Из равенств (16) следует, что приближение второй производной с помощью формулы (9) имеет различный порядок в зависимости от h в разных точках: в точках x_i , x_{i+2} имеется погрешность порядка h , а в точке x_{i+1} порядок погрешности выше ($R''_2(x_{i+1})=0$).

В случае интерполяции функции $f(x)$, имеющей на отрезке $[a, b]$ непрерывную производную до четвертого порядка включительно, мож-

но получить погрешность интерполяции второй производной, имеющую порядок h^2 и одинаковую во всех точках, с помощью многочлена Лагранжа третьей степени $L_3(x)$ по четырем узлам интерполяции x_k ($k=i, i+1, i+2, i+3$). Опуская выкладки, приведем результаты для аппроксимации второй производной:

$$\begin{aligned} y''_i &= \frac{1}{h^2} (2y_i - 5y_{i+1} + 4y_{i+2} - y_{i+3}) + O(h^2), \\ y''_{i+1} &= \frac{1}{h^2} (y_i - 2y_{i+1} + y_{i+2}) + O(h^2), \\ y''_{i+2} &= \frac{1}{h^2} (y_{i+1} - 2y_{i+2} + y_{i+3}) + O(h^2), \\ y''_{i+3} &= \frac{1}{h^2} (-y_i + 4y_{i+1} - 5y_{i+2} + 2y_{i+3}) + O(h^2). \end{aligned} \quad (18)$$

Формулы (17) дают порядок аппроксимации $O(h^2)$ для производных первого порядка функции $f(x)$, непрерывно-дифференцируемой до четвертого порядка включительно на отрезке $[a, b]$. Этот порядок сохраняется при вычислении производной второго порядка на отрезке $[a, b]$ в точках x_i ($i=0, 1, 2, \dots, n$) при $n \geq 3$ по формулам

$$\begin{aligned} y''_0 &\approx \frac{1}{h^2} (2y_0 - 5y_1 + 4y_2 - y_3), \\ y''_i &\approx \frac{1}{h^2} (y_{i-1} - 2y_i + y_{i+1}) \quad (i=1, 2, \dots, n-1), \\ y''_n &\approx \frac{1}{h^2} (-y_{n-3} + 4y_{n-2} - 5y_{n-1} + 2y_n). \end{aligned} \quad (19)$$

Если функция $f(x)$ на отрезке $[x_i, x_{i+m}]$ имеет непрерывную производную до $(m+1)$ -го порядка включительно, то справедливо представление

$$f(x) = L_m(x) + R_m(x),$$

где $L_m(x)$ — интерполяционный многочлен Лагранжа степени m , аппроксимирующий функцию $f(x)$ по узлам интерполяции $\{x_k\}$ ($k=i, i+1, \dots, i+m$); $R_m(x)$ — остаточный член интерполяционной формулы, причем

$$R_m(x) = \frac{f^{(m+1)}(\xi)}{(m+1)!} (x-x_i)(x-x_{i+1}) \dots (x-x_{i+m}), \quad \xi \in (x_i, x_{i+m}).$$

Далее, используя приведенную схему и выбирая подходящую степень m интерполяционного многочлена, можно добиться нужной точности при аппроксимации производных различных порядков.

Пример. Значения функции $y = \sin x$ определены таблицей

x	0	$\pi/6$	$\pi/3$
$\sin x$	0	0,5	0,866

Требуется с помощью формул (8) и (9) приближенно найти $y'(0)$ и $y''(0)$ и оценить погрешности результатов вычислений.

Решение. Согласно первой из формул (8), имеем

$$y'_0 \approx \frac{1}{2h} (-3y_0 + 4y_1 - y_2) = \frac{3}{\pi} (-3 \cdot 0 + 4 \cdot 0,5 - 0,866) \approx 1,05;$$

$$R'_2(0) = \frac{1}{3} \left[\frac{\pi}{6} \right]^2 f'''(\xi), \quad 0 < \xi < \frac{\pi}{3}.$$

Так как $f'(x) = \cos x$, $f''(x) = -\sin x$, $f'''(x) = -\cos x$,

$|f'''(x)| < 1$, то $|R'_2(0)| < \frac{1}{3} \left[\frac{\pi}{6} \right]^2 \approx 0,09$. Итак, $y'(0) \approx 1,05 \pm 0,09$ (точное значение $y'(0) = \cos 0 = 1$). Теперь воспользуемся формулой (9):

$$y''_0 \approx \frac{1}{h^2} (y_0 - 2y_1 + y_2) = \frac{-1 + 0,866}{\pi^2} \cdot 36 \approx -0,489,$$

$$R''_2(0) = -\frac{\pi}{6} f'''(\xi), \quad |R''_2(0)| \leq \frac{\pi}{6} \approx 0,52.$$

Как видим, для лучшей оценки производной второго порядка необходимо увеличить число узловых точек и выбрать меньший шаг. ■

Упражнения

Функция $f(x)$ определена на отрезке $[1; 1,2]$ (см. таблицу на с. 91). Выбрав шаг $h=0,05$, по формулам (17) и (19) найти приближенные значения производных $f'(x)$ и $f''(x)$ в точках 1 и 1,10; оценить погрешность вычислений. Сравнить результаты с точными значениями производных в этих точках.

ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ

§1. КВАДРАТУРНЫЕ ФОРМУЛЫ ПРЯМОУГОЛЬНИКОВ, ТРАПЕЦИИ И СИМПСОНА

Для приближенного вычисления определенного интеграла

$\int_a^b f(x) dx$ разобьем отрезок интегрирования $[a, b]$ на n равных частей

точками $x_0 = a, x_1 = x_0 + h, \dots, x_{i+1} = x_i + h, \dots, x_n = b$ (h — шаг разбиения, $h = (b-a)/n$). Значения функции $f(x)$ в точках разбиения x_i обозначим через y_i . Непрерывная подынтегральная функция

$y = f(x)$ заменяется сплайном — кусочно-полиномиальной функцией $S(x)$, аппроксимирующей данную функцию. Интегрируя функцию $S(x)$ на отрезке $[a, b]$, придем к некоторой формуле численного интегрирования (квадратурной формуле).

В зависимости от функции $S(x)$, аппроксимирующей подынтегральную функцию, будем получать различные квадратурные формулы.

Если на каждой части $[x_{i-1}, x_i]$ ($i=1, 2, \dots, n$) деления отрезка $[a, b]$ функцию $f(x)$ заменить функцией, принимающей постоянное значение, равное, например, значению функции $f(x)$ в серединной

точке i -й части $x_{i-1/2} = \frac{x_{i-1} + x_i}{2}$, то функция $S(x)$ будет иметь ступенчатый вид:

$$S(x) = S_i(x) = y_{i-1/2} = f(x_{i-1/2}), \quad x \in [x_{i-1}, x_i], \quad i=1, \dots, n.$$

В этом случае

$$\int_a^b S(x) dx = \sum_{i=1}^n \int_{x_{i-1}}^{x_i} S_i(x) dx = \sum_{i=1}^n h y_{i-1/2}$$

и получаем квадратурную формулу прямоугольников:

$$\int_a^b f(x) dx \approx \int_a^b S(x) dx = h(y_{1/2} + y_{3/2} + \dots + y_{i-1/2} + \dots + y_{n-1/2}). \quad (1)$$

Если функцию $f(x)$ на каждом отрезке $[x_{i-1}, x_i]$ заменить ее линейной интерполяцией по точкам (x_{i-1}, y_{i-1}) и (x_i, y_i) , то получим непрерывную кусочно-линейную функцию

$$S(x) = S_i(x) = y_{i-1} \frac{x_i - x}{h} + y_i \frac{x - x_{i-1}}{h}, \quad x \in [x_{i-1}, x_i], \quad i=1, \dots, n.$$

Здесь $y_i = f(x_i)$. Графиком этой функции является ломаная линия. В этом случае

$$\int_a^b S(x) dx = \sum_{i=1}^n \int_{x_{i-1}}^{x_i} S_i(x) dx = \sum_{i=1}^n h \frac{y_{i-1} + y_i}{2}$$

и получаем квадратурную формулу трапеций:

$$\int_a^b f(x) dx \approx \int_a^b S(x) dx = h \left[\frac{y_0 + y_n}{2} + y_1 + y_2 + \dots + y_{n-1} \right]. \quad (2)$$

Можно получить квадратурную формулу Симпсона, называемую также формулой парабол, если сплайн $S(x)$, аппроксимирующий подынтегральную функцию $f(x)$, представляет собой непрерывную функцию, составленную из примыкающих парабол. Потребуем, чтобы на отрезке $[x_{i-1}, x_i]$ парабола проходила через точки (x_{i-1}, y_{i-1}) , $(x_{i-1/2}, y_{i-1/2})$, (x_i, y_i) . Используя построение интерполяционного многочлена Лагранжа второго порядка на отрезке $[x_{i-1}, x_i]$, получим сплайн

$$S(x) = S_i(x) = y_{i-1} \frac{2(x-x_i)(x-x_{i-1/2})}{h^2} + y_{i-1/2} \frac{4(x_i-x)(x-x_{i-1})}{h^2} + y_i \frac{2(x-x_{i-1/2})(x-x_{i-1})}{h^2}, \quad x \in [x_{i-1}, x_i], \quad i=1, \dots, n.$$

Для дальнейших преобразований введем переменную $t \in [0, 1]$ с помощью равенства $x = x_{i-1} + ht$. Значениям t , равным 0, 1/2, 1, соответствуют значения x , равные x_{i-1} , $x_{i-1/2}$, x_i .

Выразим сплайн $S(x)$ через новую переменную t :

$$S(x) = S_i(x) = \tilde{S}_i(t) = y_{i-1}(1-t)(1-2t) + 4y_{i-1/2}t(1-t) + y_i t(2t-1) =$$

$$= y_{i-1}(1-3t+2t^2) + 4y_{i-1/2}(t-t^2) + y_i(2t^2-t) \quad (i=1, 2, \dots, n).$$

Учитывая, что $\int_0^1 (1-3t+2t^2)dt = \int_0^1 (t-t^2)dt = \int_0^1 (2t^2-t)dt = \frac{1}{6}$, имеем

$$\begin{aligned} \int_a^b S(x) dx &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} S_i(x) dx = \sum_{i=1}^n h \int_0^1 \tilde{S}_i(t) dt = \\ &= \sum_{i=1}^n \frac{h}{6} (y_{i-1} + 4y_{i-1/2} + y_i) \end{aligned}$$

и в результате приходим к квадратурной формуле парабол:

$$\int_a^b f(x) dx \approx \frac{h}{6} [y_0 + y_n + 4(y_{1/2} + y_{3/2} + \dots + y_{n-1/2}) + 2(y_1 + y_2 + \dots + y_{n-1})]. \quad (3)$$

Приближенное значение интеграла $\mathcal{Z}_{\text{параб}}$, вычисленное по квадратурной формуле парабол, можно выразить через значения $\mathcal{Z}_{\text{прял}}$ и $\mathcal{Z}_{\text{трап}}$ — результаты вычислений по квадратурным формулам прямоугольников и трапеций:

$$\mathcal{Z}_{\text{параб}} = \frac{4\mathcal{Z}_{\text{прял}} + 2\mathcal{Z}_{\text{трап}}}{6} = \frac{2}{3} \mathcal{Z}_{\text{прял}} + \frac{1}{3} \mathcal{Z}_{\text{трап}}.$$

Погрешность каждой квадратурной формулы оценивается величиной остаточного члена $R(h)$, зависящего от шага разбиения h (или от числа разбиений n):

$$R(h) = \left| \int_a^b f(x) dx - \int_a^b S(x) dx \right|.$$

Приведем оценки погрешностей квадратурных формул в том случае, когда подынтегральная функция имеет непрерывную производную второго порядка:

для формулы прямоугольников

$$R(h) \leq \frac{b-a}{24} \max_{a \leq x \leq b} |f''(x)| \cdot h^2 = \frac{(b-a)^3}{24n^2} \max_{a \leq x \leq b} |f''(x)|;$$

для формулы трапеций

$$R(h) \leq \frac{b-a}{12} \max_{a \leq x \leq b} |f''(x)| \cdot h^2 = \frac{(b-a)^3}{12n^2} \max_{a \leq x \leq b} |f''(x)|.$$

Если подынтегральная функция имеет непрерывную производную четвертого порядка, то справедлива такая оценка погрешности формулы Симпсона:

$$R(h) \leq \frac{b-a}{2880} \max_{a \leq x \leq b} |f^{(4)}(x)| \cdot h^4 = \frac{(b-a)^5}{2880n^4} \max_{a \leq x \leq b} |f^{(4)}(x)|.$$

Заметим, что при интегрировании степенной функции, степень которой не выше трех, квадратурная формула Симпсона дает точный результат.

Пример. Найти приближенные значения интеграла $\int_0^1 e^{x^2} dx$ с помощью

квадратурных формул прямоугольников, трапеций и Симпсона, если отрезок интегрирования $[0,1]$ разбит на $n = 2; 4; 10$ равных частей. Оценить величину погрешности полученных результатов в каждом случае.

Решение. Обозначим через ε погрешность результата интегрирования по квадратурным формулам (здесь $\varepsilon \sim R(h)$).

Найдем производные подынтегральной функции $f(x) = e^{x^2}$ до четвертого порядка включительно и максимальные абсолютные значения производных второго и четвертого порядков на отрезке $[0,1]$:

$$f'(x) = 2xe^{x^2}, \quad f''(x) = 2e^{x^2}(1+2x^2), \quad f'''(x) = 4xe^{x^2}(3+2x^2), \\ f^{(4)}(x) = 4e^{x^2}(3+12x^2+4x^4),$$

$$\max_{0 \leq x \leq 1} |f''(x)| = 6e, \quad \max_{0 \leq x \leq 1} |f^{(4)}(x)| = 76e.$$

При $n = 4$ получим следующие оценки величин погрешности результатов:

$$\varepsilon_{\text{прям}} = \frac{(b-a)^3}{24n^2} \max_{a \leq x \leq b} |f''(x)| = \frac{e}{64} \sim 0,0425; \quad \varepsilon_{\text{трап}} = 2\varepsilon_{\text{прям}} \sim 0,085;$$

$$\varepsilon_{\text{параб}} = \frac{(b-a)^5}{2880n^4} \max_{a \leq x \leq b} |f^{(4)}(x)| = \frac{76e}{2880 \cdot 256} \sim 0,00056.$$

Результаты вычислений по квадратурным формулам прямоугольников, трапеций и Симпсона для различных чисел разбиений n и погрешности этих результатов сведены в таблицу

Квадратурная формула	n=2		n=4		n=10	
	$\mathcal{Z}(2)$	ϵ	$\mathcal{Z}(4)$	ϵ	$\mathcal{Z}(10)$	ϵ
прямоугольников	1,40977	0,1699	1,44875	0,0425	1,46039	0,0068
трапеций	1,57158	0,3398	1,49068	0,085	1,46717	0,0136
Симпсона	1,46371	0,0045	1,46272	0,0003	1,46265	10^{-5}

Практически важно вести вычисления до достижения заданной точности ϵ по той или иной квадратурной формуле. Этой цели удовлетворяет **метод двойного пересчета**, который заключается в следующем. По квадратурной формуле проводят вычисление интеграла с шагом h и получают значение $\mathcal{Z}(h)$. Затем уменьшают шаг вдвое и получают новое приближенное значение интеграла $\mathcal{Z}(h/2)$. Чтобы определить, как сильно уклоняется значение $\mathcal{Z}(h/2)$ от точного значения интеграла $\mathcal{Z}$, используется **правило Рунге**:

$$|\mathcal{Z} - \mathcal{Z}(h/2)| \approx \frac{1}{2^{k-1}} |\mathcal{Z}(h) - \mathcal{Z}(h/2)|,$$

где $k = 2$ для формул прямоугольников и трапеций и $k = 4$ для формул Симпсона.

При заданной точности ϵ вычисления с уменьшающимся шагом проводят до окончания приближений при выполнении условия

$$\frac{1}{2^{k-1}} |\mathcal{Z}(h) - \mathcal{Z}(h/2)| < \epsilon.$$

При этом полагают $\mathcal{Z} \approx \mathcal{Z}(h/2)$ с точностью ϵ .

В Приложении (см. с. 300-305) приводятся блок-схема и программы вычисления интеграла с заданной точностью методом двойного пересчета по квадратурной формуле Симпсона. В программах используется квадратурная формула парабол следующего вида:

$$\int_a^b f(x) dx \approx \frac{h}{6} \left[y_0 - y_n + \sum_{k=1}^n (4(y_{k-1/2} + 2y_k)) \right].$$

Упражнения

Вычислить заданные интегралы по формулам прямоугольников, трапеций и Симпсона, если отрезок интегрирования разбит на $n=2$ и $n=4$ равные части. Оценить погрешность результата и сравнить приближенные значения интеграла с точными.

$$1. \int_0^1 \frac{dx}{1+x^2} \left(\mathfrak{Z} = \frac{\pi}{4} \approx 0,785 \right). \quad 2. \int_0^1 \frac{dx}{1+x} \quad (\mathfrak{Z} = \ln 2 \approx 0,693).$$

$$3. \int_0^{\pi/4} \sin 4x \, dx \quad (\mathfrak{Z} = 0,5). \quad 4. \int_0^1 \frac{dx}{\sqrt{1+x^2}} \quad (\mathfrak{Z} = \ln(1+\sqrt{2}) \approx 0,881).$$

$$5. \int_1^e \ln x \, dx \quad (\mathfrak{Z} = 1). \quad 6. \int_0^1 \ln(x+1) \, dx \quad (\mathfrak{Z} = 2\ln 2 - 1 \approx 0,386).$$

$$7. \int_0^{\pi/2} x \cos x \, dx \quad (\mathfrak{Z} = \frac{\pi}{2} - 1 \approx 0,571). \quad 8. \int_0^1 \frac{e^x dx}{1+e^{2x}} \quad (\mathfrak{Z} = \operatorname{arctg} e - \frac{\pi}{4} \approx 0,433).$$

$$9. \int_0^{\pi} \cos^3 x \, dx \quad (\mathfrak{Z} = 0). \quad 10. \int_0^{\pi/4} \frac{dx}{\cos x} \quad (\mathfrak{Z} = \ln(1+\sqrt{2}) \approx 0,881).$$

$$11. \int_0^{\pi/2} \frac{dx}{1+\sin x} \quad (\mathfrak{Z} = 1). \quad 12. \int_0^1 \operatorname{arctg} x \, dx \quad (\mathfrak{Z} = \frac{1}{4}(\pi - 2\ln 2) \approx 0,438).$$

$$13. \int_0^1 \frac{dx}{1+e^x} \quad (\mathfrak{Z} \approx 0,38). \quad 14. \int_0^{\sqrt{2}/2} \arcsin x \, dx \quad (\mathfrak{Z} = \frac{\sqrt{2}}{8}(\pi+4) - 1 \approx 0,26).$$

$$15. \int_0^{\pi/4} \operatorname{tg} x \, dx \quad (\mathfrak{Z} = \frac{1}{2} \ln 2 \approx 0,346). \quad 16. \int_{\pi/4}^{\pi/2} \operatorname{ctg} x \, dx \quad (\mathfrak{Z} = \frac{1}{2} \ln 2 \approx 0,346).$$

$$17. \int_0^1 x e^x \, dx \quad (\mathfrak{Z} = 1). \quad 18. \int_0^1 \sqrt{1+x} \, dx \quad (\mathfrak{Z} = \frac{2}{3}(2\sqrt{2}-1) \approx 1,22).$$

$$19. \int_0^{\pi/2} \frac{dx}{1 + \cos x} \quad (Z=1). \quad 20. \int_0^1 \frac{dx}{\sqrt{1+x}} \quad (Z=2(\sqrt{2}-1) \approx 0,828).$$

$$21. \int_1^e \ln^2 x \, dx \quad (Z=e-2 \approx 0,718). \quad 22. \int_0^{\pi/4} \operatorname{tg}^2 x \, dx \quad (Z=\frac{4-\pi}{4} \approx 0,215).$$

$$23. \int_{\pi/4}^{\pi/2} \operatorname{ctg}^2 x \, dx \quad (Z=\frac{4-\pi}{4} \approx 0,215). \quad 24. \int_0^{\pi/2} e^x \cos x \, dx \quad (Z=\frac{e^{\pi/2}-1}{2} \approx 1,905).$$

$$25. \int_0^{\pi/2} e^x \sin x \, dx \quad (Z = \frac{e^{\pi/2}+1}{2} \approx 2,905).$$

§2. КВАДРАТУРНЫЕ ФОРМУЛЫ ГАУССА

Пусть отрезок интегрирования $[a, b]$ непрерывной функции $f(x)$ разбит на n равных частей точками $x_0=a, x_1, \dots, x_{i-1}, x_i, \dots, x_n=b$ (n — шаг разбиения, $h=(b-a)/n$). Обозначим через $S(x)$ сплайн-функцию, аппроксимирующую подынтегральную функцию $f(x)$.

Пусть на каждой части разбиения $[x_{i-1}, x_i]$ ($i=1, \dots, n$) расположено m узлов $(x_{i1}, x_{i2}, \dots, x_{im})$, в которых подынтегральная функция принимает значения $f(x_{ij})$ ($j=1, \dots, m$). Предположим, что функция $f(x)$ на каждой i -й части аппроксимируется многочленом $S_i(x)$ степени p , $x \in [x_{i-1}, x_i]$ ($i=1, \dots, n$). При этом на многочлен $S_i(x)$ накладываются два ограничения:

- а) значения многочлена и подынтегральной функции равны в узлах интерполяции, т.е. $S_i(x_{ij})=f(x_{ij})$ ($i=1, \dots, n$; $j=1, \dots, m$);
- б) определенный интеграл от функции $S_i(x)$ на отрезке $[x_{i-1}, x_i]$ выражается через значения подынтегральной функции $f(x_{ij})$ в узлах в виде их линейной комбинации, т.е.

$$\int_{x_{i-1}}^{x_i} S_i(x) dx = C_1 f(x_{i1}) + C_2 f(x_{i2}) + \dots + C_m f(x_{im}). \quad (4)$$

Квадратурные формулы Гаусса для выбранной степени p сплайна будут определены, если из условий а) и б) удастся найти m неизвестных коэффициентов C_j и координаты m узлов x_{ij} ($j=1, \dots, m$).

Задача решается одновременно для всех n частей разбиения отрезка $[a, b]$, если выразить $x \in [x_{i-1}, x_i]$ ($i=1, \dots, n$) через переменную $t \in [-1, 1]$ так:

$$x = \frac{x_i - x_{i-1}}{2} t + \frac{x_i + x_{i-1}}{2}; \quad dx = \frac{x_i - x_{i-1}}{2} dt = \frac{h}{2} dt.$$

Положим $\tilde{f}_i(t) = f\left[\frac{x_i - x_{i-1}}{2} t + \frac{x_i + x_{i-1}}{2}\right]$; $\tilde{S}_i(t) = S_i(x(t))$; $C_i = \frac{h}{2} \tilde{C}_i$.

Тогда $f(x_{ij}) = \tilde{f}_i(t_j)$ ($i=1, \dots, n$; $j=1, \dots, m$) и соотношение (4) перепишем в виде

$$\int_{-1}^1 \tilde{S}_i(t) dt = \tilde{C}_1 \tilde{f}_i(t_1) + \tilde{C}_2 \tilde{f}_i(t_2) + \dots + \tilde{C}_m \tilde{f}_i(t_m). \quad (5)$$

Выведем квадратурную формулу Гаусса с тремя узлами ($m=3$). Для этого необходимо определить шесть величин: $\tilde{C}_1, \tilde{C}_2, \tilde{C}_3, t_1, t_2, t_3$. Функция $\tilde{S}_i(t)$ — многочлен степени p , общий вид которого

$$\tilde{S}_i(t) = a_0 + a_1 t + a_2 t^2 + \dots + a_p t^p. \quad (6)$$

Подставив соотношение (6) в (5) и учитывая, что $\tilde{f}_i(t_j) = \tilde{S}_i(t_j)$ ($j=1, 2, 3$), получим тождество относительно коэффициентов a_k ($k=0, 1, 2, \dots, p$):

$$\int_{-1}^1 (a_0 + \dots + a_p t^p) dt = \tilde{C}_1 (a_0 + \dots + a_p t_1^p) + \tilde{C}_2 (a_0 + \dots + a_p t_2^p) + \tilde{C}_3 (a_0 + \dots + a_p t_3^p).$$

Шесть неизвестных будут определены однозначно из системы шести уравнений. В общем случае степень p аппроксимирующего многочлена

всегда является нечетным числом и связана с числом узлов m соотношением $p = 2m-1$. В частности, для трех узлов имеем многочлен пятой степени ($p=5$).

Множители при a_k в левой части тождества вычисляем так:

$$\int_{-1}^1 t^k dt = \frac{t^{k+1}}{k+1} \Big|_{-1}^1 = \frac{1 - (-1)^{k+1}}{k+1} = \begin{cases} \frac{2}{k+1}, & \text{если } k - \text{четное;} \\ 0, & \text{если } k - \text{нечетное.} \end{cases}$$

Приравняем подобные выражения в левой и правой частях тождества при одинаковых коэффициентах $a_0, a_1, \dots, a_5$. Получим следующие шесть уравнений:

$$\begin{cases} \tilde{C}_1 + \tilde{C}_2 + \tilde{C}_3 = 2, & \tilde{C}_1 t_1 + \tilde{C}_2 t_2 + \tilde{C}_3 t_3 = 0, \\ \tilde{C}_1 t_1^2 + \tilde{C}_2 t_2^2 + \tilde{C}_3 t_3^2 = \frac{2}{3}, & \tilde{C}_1 t_1^3 + \tilde{C}_2 t_2^3 + \tilde{C}_3 t_3^3 = 0, \\ \tilde{C}_1 t_1^4 + \tilde{C}_2 t_2^4 + \tilde{C}_3 t_3^4 = \frac{2}{5}, & \tilde{C}_1 t_1^5 + \tilde{C}_2 t_2^5 + \tilde{C}_3 t_3^5 = 0. \end{cases} \quad (7)$$

Эту систему позволяет упростить следующее свойство ее решения: неизвестные t_i ($i=1, 2, \dots, m$) системы $2m$ уравнений вида

$$C_1 t_1^k + C_2 t_2^k + \dots + C_m t_m^k = \begin{cases} \frac{2}{k+1}, & \text{если } k - \text{четное;} \\ 0, & \text{если } k - \text{нечетное} \end{cases}$$

(где $k=0, 1, \dots, 2m-1$) являются нулями многочлена Лежандра

$$P_m(t) = \frac{1}{2^m m!} \frac{d^m}{dt^m} (t^2 - 1)^m; \text{ нули многочлена принадлежат интервалу}$$

$(-1, 1)$ и расположены симметрично относительно середины интервала.

В рассматриваемом частном случае $m=3$ и $P_3(t) = \frac{1}{2}(5t^3 - 3t)$.

Находим нули многочлена из уравнения $5t^3 - 3t = 0$. Подставив корни

$$\text{уравнения } t_1 = -\sqrt{\frac{3}{5}}, \quad t_2 = 0, \quad t_3 = \sqrt{\frac{3}{5}} \text{ в (7), получим систему}$$

трех линейных уравнений относительно переменных $\tilde{C}_1, \tilde{C}_2, \tilde{C}_3$:

$$\tilde{C}_1 + \tilde{C}_2 + \tilde{C}_3 = 2, \quad -\tilde{C}_1 \cdot \sqrt{\frac{3}{5}} + \tilde{C}_3 \cdot \sqrt{\frac{3}{5}} = 0, \quad \tilde{C}_1 \cdot \frac{3}{5} + \tilde{C}_3 \cdot \frac{3}{5} = \frac{2}{3}.$$

Очевидное решение этой системы есть $\tilde{C}_1 = \tilde{C}_3 = 5/9$ и $\tilde{C}_2 = 8/9$.

Теперь подставим найденные значения в соотношение (4):

$$\int_{x_{i-1}}^{x_i} S_i(x) dx = \frac{h}{18} (5f(x_{i1}) + 8f(x_{i2}) + 5f(x_{i3})),$$

где $x_{i2} = \frac{x_i + x_{i-1}}{2}$, $x_{i1} = x_{i2} - \sqrt{\frac{3}{5}} \cdot \frac{h}{2}$, $x_{i3} = x_{i2} + \sqrt{\frac{3}{5}} \cdot \frac{h}{2}$.

Итак, квадратурная формула Гаусса с тремя узлами записывается в виде

$$\int_a^b f(x) dx \approx \int_a^b S(x) dx = \sum_{i=1}^n \int_{x_{i-1}}^{x_i} S_i(x) dx = \sum_{i=1}^n \frac{h}{18} (5f(x_{i1}) + 8f(x_{i2}) + 5f(x_{i3})).$$

Если подынтегральная функция имеет непрерывную производную шестого порядка, то для оценки погрешности формулы Гаусса с тремя узлами можно использовать неравенство

$$R(h) \leq \frac{b-a}{2016000} \max_{a \leq x \leq b} |f^{(6)}(x)| \cdot h^6 = \frac{(b-a)^7}{2016000n^6} \max_{a \leq x \leq b} |f^{(6)}(x)|.$$

При вычислении интеграла до достижения заданной точности ε методом двойного пересчета условие окончания вычислений имеет вид

$$\frac{1}{2^{k-1}} |\mathfrak{I}(h) - \mathfrak{I}(h/2)| < \varepsilon,$$

где $k=2m$ (m — число узлов в квадратурной формуле Гаусса). Полагают, что $\mathfrak{I} \approx \mathfrak{I}(h/2)$ с точностью ε .

Пример. Найти приближенное значение интеграла $\int_0^1 e^{x^2} dx$ по квадратурной формуле Гаусса с тремя узлами для $n=1$ (без разбиения отрезка $[0,1]$ на части, $h=1$). Сравнить полученный результат и погрешность с результатами вычислений примера из §1.

Решение. Найдем производные подынтегральной функции $f(x)=e^{x^2}$ до шестого порядка включительно, продолжая вычисления примера из §1:

$$f^{(4)}(x) = 4e^{x^2}(3+12x^2+4x^4), \quad f^{(5)}(x) = 8e^{x^2}(15x+20x^3+4x^5),$$

$$f^{(6)}(x) = 8e^{x^2}(15+90x^2+60x^4+8x^6), \quad \max_{0 \leq x \leq 1} |f^{(6)}(x)| = 1384e.$$

Тогда

$$R(n) \leq \frac{b-a}{2016000} \max_{0 \leq x \leq 1} |f^{(6)}(x)| \cdot h^6 = \frac{1384e}{2016000} \approx 0,0019.$$

С погрешностью, не большей чем 0,0019, имеем

$$\int_0^1 e^{x^2} dx \approx \frac{1}{18} (5f(x_1) + 8f(x_2) + 5f(x_3)) = 1,46241.$$

Здесь $x_2 = 0,5$;

$$f(x_2) = 1,284025;$$

$$x_1 = x_2 - \frac{1}{2} \sqrt{\frac{3}{5}} = 0,112702; \quad f(x_1) = 1,012783;$$

$$x_3 = x_2 + \frac{1}{2} \sqrt{\frac{3}{5}} = 0,887298; \quad f(x_3) = 2,19745. \blacksquare$$

Упражнения

Вычислить приближенные значения интегралов 1-25 (см. упражнения к §1), используя квадратурную формулу Гаусса с тремя узлами для числа разбиения отрезка интегрирования $n=1$. Оценить погрешность результата; сравнить приближенные значения интеграла со значениями, полученными в упражнениях к §1, и с их точными значениями.

§3. ПРИБЛИЖЕННОЕ ВЫЧИСЛЕНИЕ НЕСОБСТВЕННЫХ ИНТЕГРАЛОВ С БЕСКОНЕЧНЫМИ ПРЕДЕЛАМИ

Определение. Пусть функция $f(x)$ непрерывна на промежутке $[a, \infty)$ и существует предел интеграла как функции верхнего предела интегрирования:

$$\lim_{x \rightarrow \infty} \int_a^x f(t) dt. \quad (8)$$

Тогда этот предел называют **несобственным интегралом** функции $f(x)$

на промежутке $[a, \infty)$ и обозначают так: $\int_a^\infty f(x) dx$.

В случае существования предела (8) говорят, что несобственный интеграл функции $f(x)$ сходится на промежутке $[a, \infty)$.

Представим несобственный интеграл на промежутке $[a, \infty)$ в виде суммы определенного интеграла на отрезке $[a, b]$ и несобственного на промежутке (b, ∞) :

$$\int_a^{\infty} f(x) dx = \int_a^b f(x) dx + \int_b^{\infty} f(x) dx.$$

Интеграл функции $f(x)$ сходится на $[a, \infty)$, если для любого числа $\varepsilon > 0$ существует число b такое, что абсолютная величина второго слагаемого будет меньше ε , т.е.

$$\left| \int_b^{\infty} f(x) dx \right| < \varepsilon. \quad (9)$$

Значение сходящегося несобственного интеграла на промежутке (a, ∞) равно с точностью ε определенному интегралу от функции $f(x)$ на отрезке $[a, b]$, если в соответствии с выбранным ε удастся найти значение верхнего предела b определенного интеграла, удовлетворяющего условию (9).

Пример 1. Даны сходящиеся несобственные интегралы:

$$1) \int_a^{\infty} \frac{C dx}{x^p}, \quad a > 0, \quad p > 1; \quad 2) \int_a^{\infty} C e^{-\lambda x} dx, \quad \lambda > 0; \quad C > 0.$$

Используя условие (9), аппроксимировать их определенными интегралами с точностью ε . Осуществить замену переменной интегрирования так, чтобы верхний предел интегрирования b был равен $a+10$. Замена переменной интегрирования имеет смысл в случаях, когда условие (6) дает большой отрезок интегрирования $[a, b]$, малоудобный для численного интегрирования.

Решение. Подынтегральные функции в обоих случаях знакоположи-

тельны; тогда условие (9) примет вид $\int_b^{\infty} f(x) dx < \varepsilon$.

1) Имеем

$$\int_b^{\infty} \frac{C dx}{x^p} = \frac{C x^{-p+1}}{-p+1} \Big|_b^{\infty} = -\frac{C}{(p-1)x^{p-1}} \Big|_b^{\infty} = \frac{C}{(p-1)b^{p-1}} < \varepsilon.$$

Таким образом,

$$b > \left[\frac{C}{(p-1)\varepsilon} \right]^{1/(p-1)} \quad (10)$$

и в качестве верхнего предела удобно принять наименьшее целое число, удовлетворяющее неравенству (10). В частности, если $a=C=1$, $p=2$ и $\varepsilon=0,001$, то неравенство (10) дает $b>1000$. Окончательно в этом частном случае получим

$$\int_1^{\infty} \frac{dx}{x^2} \approx \int_1^{1001} \frac{dx}{x^2} = -\frac{1}{x} \Big|_1^{1001} = 1 - \frac{1}{1001},$$

т.е. найдено значение несобственного интеграла с точностью, не меньшей чем 0,001 (его точное значение равно 1).

Простейшая замена переменной интегрирования, сокращающая длину отрезка интегрирования, такова: $x = t^m$. Очевидно, что

$$\int_a^b \frac{Cdx}{x^p} = \int_{a_1}^{b_1} \frac{Cmt^{m-1}dt}{t^{pm}} = \int_{a_1}^{b_1} \frac{Cm dt}{t^{m(p-1)+1}}, \quad \text{где } a_1 = \sqrt[m]{a}, \quad b_1 = \sqrt[m]{b},$$

и показатель степени m полагаем равным ближайшему целому числу, не меньшему чем $m = \ln b / \ln b_1$.

В рассматриваемом частном случае $b = 1001$, $b_1 = a+10=11$ и $m = \ln 1001 / \ln 11 \approx 3$. Сделав замену переменной $x = t^3$, получаем

$$\int_1^{\infty} \frac{dx}{x^2} \approx \int_1^{11} \frac{3dt}{t^4} = -\frac{1}{t^3} \Big|_1^{11} = 1 - \frac{1}{11^3} = 1 - \frac{1}{1331},$$

что отличается от точного значения на величину второго слагаемого.

2) Имеем

$$\int_b^{\infty} C e^{-\lambda x} dx = -\frac{C}{\lambda} e^{-\lambda x} \Big|_b^{\infty} = \frac{C}{\lambda} e^{-\lambda b} < \varepsilon.$$

Таким образом,

$$b > \frac{1}{\lambda} \ln \frac{C}{\lambda \varepsilon}, \quad (11)$$

и в расчетах можно считать b равным наименьшему целому числу, удовлетворяющему неравенству (8). Если положить $a = 0$, $C=\lambda=1$ $\varepsilon = 0,001$, то $b > 3 \ln 10 \approx 6,9$ и

$$\int_0^{\infty} e^{-x} dx \approx \int_0^7 e^{-x} dx = -e^{-x} \Big|_0^7 = 1 - e^{-7} = 1 - 0,000912.$$

Точное значение несобственного интеграла равно 1. ■

Несобственные интегралы примера 1 можно считать эталонными для встречающихся в практике интегралов. Ниже на примерах некоторых абсолютно сходящихся интегралов мы покажем, как используются эталонные интегралы.

Определение. Несобственный интеграл функции $f(x)$ на $[a, \infty)$ называют **абсолютно сходящимся**, если сходится несобственный интеграл абсолютной величины функции $|f(x)|$ на этом промежутке.

Если $|g(x)| \leq |f(x)|$ для любых значений x на промежутке $[a, \infty)$ или функция $|g(x)|$ эквивалентна $|f(x)|$ при $x \rightarrow \infty$ (предел отношения этих функций равен единице при x , стремящемся к бесконечности) и интеграл функции $|f(x)|$ сходится на $[a, \infty)$, то несобственный интеграл функции $g(x)$ также сходится на $[a, \infty)$.

Приведенная форма достаточных условий сходимости абсолютно сходящихся интегралов позволяет в неравенстве (9) использовать упрощенные подынтегральные функции вместо заданных.

В упражнениях к этому параграфу удобно сравнивать заданную подынтегральную функцию с эталонной, если при больших значениях аргумента применять следующие соотношения:

- 1) $Ax^{\alpha} + Bx^{\beta} \sim Ax^{\alpha}$, если $\alpha > 0$, $\alpha > \beta$, A, B — числа;
- 2) $\ln x < x^p$, p — любое положительное число.

Пример 2. Даны несобственные интегралы:

$$1) \int_0^{\infty} \frac{\sin x}{\sqrt{x+1} (2 + \sqrt{x^3})} dx; \quad 2) \int_0^{\infty} e^{-x^2} \ln(x+1) dx.$$

Аппроксимировать их определенными интегралами с точностью, не меньшей чем $\epsilon = 0,001$.

Решение. Прежде чем искать верхние пределы интегрирования с помощью неравенства (9), упростим подынтегральные функции.

1) Имеем

$$\frac{|\sin x|}{\sqrt{x+1} (2 + \sqrt{x^3})} \leq \frac{1}{\sqrt{x+1} (2 + \sqrt{x^3})} \leq \frac{1}{\sqrt{x} \cdot \sqrt{x^3}} = \frac{1}{x^2}, \quad x > 0.$$

Теперь воспользуемся неравенством (9):

$$\left| \int_b^{\infty} \frac{\sin x}{\sqrt{x+1}(2+\sqrt{x^3})} dx \right| \leq \int_b^{\infty} \frac{|\sin x|}{\sqrt{x+1}(2+\sqrt{x^3})} dx \leq \int_b^{\infty} \frac{dx}{x^2} < \varepsilon = 0,001.$$

Из решения примера 1 следует, что $b=1001$, $b_1=11$ при замене $x=t^3$

$$\text{и тогда } \int_1^{\infty} \frac{\sin x}{\sqrt{x+1}(2+\sqrt{x^3})} dx \approx \int_1^{1001} \frac{\sin x}{\sqrt{x+1}(2+\sqrt{x^3})} dx \approx \int_1^{11} \frac{3t^2 \sin t^3}{\sqrt{t^3+1}(2+\sqrt{t^9})} dt$$

с точностью, не меньшей чем 0,001.

2) Так как $e^{-x^2} \ln(x+1) \leq e^{-x^2} \ln(2x) \leq 2xe^{-x^2}$, $x > 1$, то

$$\int_b^{\infty} e^{-x^2} \ln(x+1) dx \leq \int_b^{\infty} e^{-x^2} \ln(2x) dx \leq \int_b^{\infty} 2xe^{-x^2} dx = \int_{b^2}^{\infty} e^{-u} du < \varepsilon = 0,001,$$

где $u = x^2$. В примере 1 было получено, что $b^2=7$, $b=\sqrt{7}$.

$$\text{Окончательно находим } \int_0^{\infty} e^{-x^2} \ln(x+1) dx \approx \int_0^{\sqrt{7}} e^{-x^2} \ln(x+1) dx.$$

Замечание 1. Естественно, что окончательная погрешность при вычислении несобственного интеграла равна сумме погрешности ε его представления в виде определенного интеграла и погрешности при вычислении определенного интеграла по одной из квадратурных формул. Можно полагать, например, погрешность вычисления определенного интеграла равной $\varepsilon/10$.

Замечание 2. Несобственные интегралы на промежутке $(-\infty, a]$ вычисляются аналогично изложенному.

Упражнения

Используя определение, показать, что несобственный интеграл

$$\int_a^{\infty} f(x) dx \quad \text{сходится. Аппроксимировать его определенным}$$

интегралом с точностью $\varepsilon = 10^{-3}$. Вычислить определенный интеграл с помощью квадратурной формулы Гаусса методом двойного

пересчета с точностью 10^{-4} . Сравнить полученный результат с точным значением несобственного интеграла, вычисленным с шестью верными знаками.

$$1. \int_1^{\infty} \frac{dx}{x \sqrt{x^4 + 1}} \quad (Z = \frac{1}{2} \ln(1 + \sqrt{2})). \quad 2. \int_0^{\infty} e^{-x} \sin x \, dx \quad (Z = 0,5).$$

$$3. \int_1^{\infty} \frac{dx}{x \sqrt{x^6 + 1}} \quad (Z = \frac{1}{3} \ln(1 + \sqrt{2})). \quad 4. \int_0^{\infty} e^{-x} \cos x \, dx \quad (Z = 0,5).$$

$$5. \int_0^{\infty} e^{-x^2} \sin^2 x \, dx \quad (Z = \frac{\sqrt{\pi}}{4} (1 - e^{-1})). \quad 6. \int_0^{\infty} x e^{-x} \sin x \, dx \quad (Z = 0,5).$$

$$7. \int_0^{\infty} e^{-x^2} \cos^2 x \, dx \quad (Z = \frac{\sqrt{\pi}}{4} (1 + e^{-1})). \quad 8. \int_0^{\infty} x e^{-x} \cos x \, dx \quad (Z = 0).$$

$$9. \int_0^{\infty} \frac{dx}{1 + e^{2x}} \quad (Z = \frac{1}{2} \ln 2). \quad 10. \int_0^{\infty} x e^{-x} \, dx \quad (Z = 1).$$

$$11. \int_{\sqrt{3}}^{\infty} \frac{dx}{x \sqrt{(x^2 + 1)^3}} \quad (Z = \frac{\ln 3 - 1}{2}). \quad 12. \int_0^{\infty} e^{-x^2} \, dx \quad (Z = \frac{\sqrt{\pi}}{2}).$$

$$13. \int_0^{\infty} \frac{dx}{(x + \sqrt{x^2 + 1})^3} \quad (Z = \frac{3}{8}). \quad 14. \int_0^{\infty} x^2 e^{-2x} \, dx \quad (Z = 0,25).$$

$$15. \int_0^{\infty} \frac{dx}{(x + \sqrt{x^2 + 1})^4} \quad (Z = \frac{4}{15}). \quad 16. \int_0^{\infty} \frac{dx}{1 + \sqrt{e^x}} \quad (Z = 2 \ln 2).$$

17. $\int_0^{\infty} x^2 e^{-3x} dx \quad (\mathfrak{Z} = \frac{2}{27}).$ 18. $\int_0^{\infty} \frac{x dx}{(1+x^2)^2} \quad (\mathfrak{Z} = 0,5).$
19. $\int_0^{\infty} \frac{x \sin x dx}{(1+x^2)^3} \quad (\mathfrak{Z} = \frac{\pi}{8e}).$ 20. $\int_1^{\infty} \frac{\ln x dx}{x^4} \quad (\mathfrak{Z} = \frac{1}{9}).$
21. $\int_0^{\infty} \frac{x dx}{1+x^4} \quad (\mathfrak{Z} = \frac{\pi}{4}).$ 22. $\int_1^{\infty} \frac{\ln^2 x dx}{x^4} \quad (\mathfrak{Z} = \frac{2}{27}).$
23. $\int_0^{\infty} \frac{dx}{(1+x^2) \operatorname{ch} \frac{\pi x}{2}} \quad (\mathfrak{Z} = \ln 2).$ 24. $\int_0^{\infty} \frac{dx}{1+x^3} \quad (\mathfrak{Z} = \frac{2\pi\sqrt{3}}{9}).$
25. $\int_0^{\infty} \frac{dx}{(1+x^2) \operatorname{ch} \pi x} \quad (\mathfrak{Z} = 2 - \frac{\pi}{2}).$

§4. ПРИБЛИЖЕННОЕ ВЫЧИСЛЕНИЕ НЕСОБСТВЕННЫХ ИНТЕГРАЛОВ ОТ ФУНКЦИИ С БЕСКОНЕЧНЫМ РАЗРЫВОМ

Пусть функция $f(x)$ непрерывна на промежутке (a, b) и предел функции при x , стремящемся к b , равен бесконечности (бесконечный разрыв функции в точке b), т.е.

$$\lim_{\substack{x \rightarrow b \\ (x < b)}} f(x) = \lim_{x \rightarrow b-0} f(x) = \infty,$$

или не существует.

Определение. Если существует предел интеграла как функции верхнего предела интегрирования:

$$\lim_{\substack{x \rightarrow b \\ (x < b)}} \int_a^x f(t) dt, \quad (12)$$

то этот предел называют **несобственным интегралом** функции $f(x)$ на отрезке $[a, b]$ и обозначают, как и определенный интеграл, в ви-

$$\text{де} \quad \int_a^b f(x) dx:$$

В случае существования предела (12) говорят, что несобственный интеграл функции $f(x)$ сходится на отрезке $[a, b]$.

Аналогично определяется несобственный интеграл на отрезке $[a, b]$ для функции, имеющей бесконечный разрыв в точке a .

Несобственный интеграл функции, имеющей бесконечный разрыв в некоторой внутренней точке $c \in (a, b)$, определяют на отрезке $[a, b]$ как сумму сходящихся интегралов на отрезках $[a, c]$ и $[c, b]$.

Несобственные интегралы с бесконечным разрывом подынтегральной функции на отрезке интегрирования с помощью замены переменной интегрирования преобразуют к несобственным интегралам с бесконечными пределами. Покажем это на примерах.

Пример. Даны несобственные интегралы от разрывных функций на отрезках интегрирования:

$$1) \int_0^1 \sin \frac{1}{x} dx; \quad 2) \int_0^1 \frac{dx}{(2+x)\sqrt{1-x^2}}.$$

С помощью замены переменной интегрирования преобразовать их в несобственные интегралы с бесконечными пределами. Аппроксимировать эти несобственные интегралы определенными интегралами с точностью, не меньшей чем $\epsilon = 0,001$.

Решение. 1) Подынтегральная функция $\sin \frac{1}{x}$ в точке $x=0$ (нижнем пределе интегрирования) не определена и $\lim_{x \rightarrow 0} \sin \frac{1}{x}$ не существует.

Произведем замену переменной так, чтобы особой точке $x=0$ соответствовала бесконечно удаленная. Простейшая замена переменной интегрирования есть $x = \frac{1}{t}$. Очевидно, что если $t \rightarrow \infty$, то $x \rightarrow 0$, а если $x = 1$, то $t = 1$; $dx = -\frac{1}{t^2} dt$. Тогда

$$\int_0^1 \sin \frac{1}{x} dx = - \int_{\infty}^1 \frac{\sin t dt}{t^2} = \int_1^{\infty} \frac{\sin t dt}{t^2}.$$

Применяя практически те же преобразования, что и в примере 2(1) из §3, для последнего несобственного интеграла на промежутке $[1, \infty)$ находим его приближение определенным на отрезке $[1, 1001]$. Далее, после замены переменной $t = u^3$ получаем определенный интеграл на отрезке $[1, 11]$, аппроксимирующий заданный несобственный интеграл с пог-

решностью, не большей чем 0,001:

$$\int_0^1 \sin \frac{1}{x} dx = \int_1^{\infty} \frac{\sin t dt}{t^2} \approx \int_1^{1001} \frac{\sin t dt}{t^2} \approx \int_1^{11} \frac{3 \sin u^3 du}{u^4}.$$

2) Знаменатель подынтегральной функции $(2+x)\sqrt{1-x^2}$ обращается в нуль при $x=1$. Следовательно, подынтегральная функция при $x \rightarrow 1$ ($x < 1$) стремится к бесконечности. Несобственный интеграл на отрезке $[0,1]$ с помощью замены переменной $1-x = \frac{1}{t}$ преобразуется в несобственный интеграл на промежутке $(0, \infty)$. Если $t \rightarrow \infty$, то $x \rightarrow 1$; если же $x=0$, то $t=1$; $dx = -\frac{1}{t^2} dt$. Выполним последовательно преобразования:

$$\begin{aligned} \int_0^1 \frac{dx}{(2+x)\sqrt{1-x^2}} &= \int_0^1 \frac{dx}{(2+x)\sqrt{(1-x)(1+x)}} = \\ &= \int_1^{\infty} \frac{dt}{t^2 \left(3 - \frac{1}{t}\right) \sqrt{\frac{1}{t} \left(2 - \frac{1}{t}\right)}} = \int_1^{\infty} \frac{dt}{(3t-1)\sqrt{2t-1}}. \end{aligned}$$

Несобственный интеграл на промежутке $[1, \infty)$ аппроксимируется определенным на отрезке $[1, b]$. Значение верхнего предела b находится из неравенства (9) после упрощений подынтегральной функции:

$$\int_b^{\infty} \frac{dt}{(3t-1)\sqrt{2t-1}} \leq \int_b^{\infty} \frac{dt}{2t\sqrt{t}} = -\frac{1}{\sqrt{t}} \Big|_b^{\infty} = \frac{1}{\sqrt{b}} < \varepsilon = 0,001.$$

Значение верхнего предела, удовлетворяющего неравенству $b > \varepsilon^2 = 10^6$, велико и для вычислений произведем еще одну замену переменной $t = u^6$. Дальнейшие преобразования приводят к определенному интегралу на отрезке $[1, 11]$:

$$\int_0^1 \frac{dx}{(2+x)\sqrt{1-x^2}} = \int_1^{\infty} \frac{dt}{(3t-1)\sqrt{2t-1}} \approx \int_1^{10^6} \frac{dt}{(3t-1)\sqrt{2t-1}} \approx \int_1^{11} \frac{6u^5 du}{(3u^6-1)\sqrt{2u^6-1}}.$$

Приведем окончательные результаты (определенные интегралы вычис-

лены по квадратурной формуле Гаусса с точностью $\varepsilon = 0,000001$):

$$\int_0^1 \sin \frac{1}{x} dx \approx \int_1^{11} \frac{3 \sin u^3 du}{u^4} \approx 0,504067;$$

$$\int_0^1 \frac{dx}{(2+x)\sqrt{1-x^2}} \approx \int_1^{11} \frac{6u^5 du}{(3u^6-1)\sqrt{2u^6-1}} \approx 0,604244. \blacksquare$$

Упражнения

Показать, что несобственный интеграл $\int_a^b f(x)dx$ от неограни-

ченной на одном из концов отрезка $[a, b]$ функции $f(x)$ сходится. Используя замену переменной интегрирования, преобразовать несобственный интеграл от разрывной на отрезке интегрирования функции в несобственный интеграл с бесконечным пределом. Аппроксимировать несобственный интеграл определенным с точностью $\varepsilon = 10^{-3}$. Вычислить определенный интеграл с помощью квадратурной формулы Гаусса методом двойного пересчета с точностью 10^{-4} . Сравнить полученный результат с его аналитическим значением, вычисленным с шестью верными знаками.

$$1. \int_0^1 \frac{dx}{\sqrt{x}(1+x)} \quad (Z = \frac{\pi}{2}). \quad 2. \int_0^1 \frac{dx}{\sqrt{x}(3+x)} \quad (Z = \frac{\pi\sqrt{3}}{9}).$$

$$3. \int_0^3 \frac{dx}{\sqrt{x}(1+x)} \quad (Z = \frac{2\pi}{3}). \quad 4. \int_0^3 \frac{dx}{\sqrt{x}(3+x)} \quad (Z = \frac{\pi\sqrt{3}}{6}).$$

$$5. \int_0^{0,25} \frac{dx}{\sqrt{x}(1-x)} \quad (Z = \ln 3). \quad 6. \int_{0,5}^1 \frac{(x+2)dx}{\sqrt{1-x}} \quad (Z = \frac{17\sqrt{2}}{6}).$$

$$7. \int_0^1 \frac{(x+2)dx}{\sqrt{1-x}} \quad (\mathfrak{I} = \frac{16}{3}). \quad 8. \int_1^2 \frac{x dx}{\sqrt{2-x}} \quad (\mathfrak{I} = \frac{10}{3}).$$

$$9. \int_0^2 \frac{(x+1)dx}{\sqrt{2-x}} \quad (\mathfrak{I} = \frac{14\sqrt{2}}{3}). \quad 10. \int_{-0,5}^0 \frac{x dx}{\sqrt{1+2x}} \quad (\mathfrak{I} = -\frac{1}{3}).$$

$$11. \int_{-1}^0 \frac{dx}{(x+2)\sqrt{1+x}} \quad (\mathfrak{I} = \frac{\pi}{2}). \quad 12. \int_0^{0,5} \frac{x dx}{\sqrt{1-2x}} \quad (\mathfrak{I} = \frac{1}{3}).$$

$$13. \int_{-1}^0 \frac{dx}{(x+4)\sqrt{1+x}} \quad (\mathfrak{I} = \frac{\pi\sqrt{3}}{9}). \quad 14. \int_0^{0,5} \frac{(x+1)dx}{\sqrt{1-2x}} \quad (\mathfrak{I} = \frac{4}{3}).$$

$$15. \int_{-1}^0 \frac{dx}{(3-x)\sqrt{1+x}} \quad (\mathfrak{I} = \frac{\ln 3}{2}). \quad 16. \int_{0,5}^1 \frac{(x+2)dx}{\sqrt{2x-1}} \quad (\mathfrak{I} = \frac{8}{3}).$$

$$17. \int_{-1}^0 \frac{(x+2)dx}{\sqrt{1+x}} \quad (\mathfrak{I} = \frac{8}{3}). \quad 18. \int_1^2 \frac{x dx}{(x+2)\sqrt{x-1}} \quad (\mathfrak{I} = \frac{18-2\pi\sqrt{3}}{9}).$$

$$19. \int_1^3 \frac{x dx}{\sqrt{3-x}} \quad (\mathfrak{I} = \frac{14\sqrt{2}}{3}). \quad 20. \int_0^3 \frac{x dx}{\sqrt{3-x}} \quad (\mathfrak{I} = 4\sqrt{3}).$$

$$21. \int_0^4 \frac{x dx}{\sqrt{4-x}} \quad (\mathfrak{I} = \frac{32}{3}). \quad 22. \int_1^2 \frac{(x+1)dx}{\sqrt{x-1}} \quad (\mathfrak{I} = \frac{14}{3}).$$

$$23. \int_{-0,5}^0 \frac{dx}{\sqrt{-x}(1+x)} \quad (\mathfrak{I} = \ln \left[\frac{\sqrt{2}+1}{\sqrt{2}-1} \right]).$$

$$24. \int_0^1 \frac{dx}{\sqrt{x}(3-x)} \quad (\mathfrak{I} = \frac{\sqrt{3}}{3} \ln \left[\frac{\sqrt{3}+1}{\sqrt{3}-1} \right]).$$

$$25. \int_{-1}^0 \frac{x dx}{(x-1)\sqrt{x+1}} \quad (\mathfrak{I} = 2 - \frac{\sqrt{2}}{2} \ln \left[\frac{\sqrt{2}+1}{\sqrt{2}-1} \right]).$$

$$26. \int_{-1}^0 \frac{dx}{(x-1)\sqrt{x+1}} \quad (\mathfrak{I} = -\frac{\sqrt{2}}{2} \ln \left[\frac{\sqrt{2}+1}{\sqrt{2}-1} \right]).$$

$$27. \int_0^1 \frac{dx}{(x+3)\sqrt{1-x}} \quad (\mathfrak{I} = \frac{1}{2} \ln 3).$$

$$28. \int_0^1 \frac{dx}{\sqrt{x}(5-x)} \quad (\mathfrak{I} = \frac{\sqrt{5}}{5} \ln \left[\frac{\sqrt{5}+1}{\sqrt{5}-1} \right]).$$

$$29. \int_0^1 \frac{(x+1)dx}{\sqrt{x}(x+3)} \quad (\mathfrak{I} = \frac{2\sqrt{3}}{9} (3\sqrt{3} - \pi)).$$

$$30. \int_1^2 \frac{(x+2)dx}{\sqrt{x-1}} \quad (\mathfrak{I} = \frac{20}{3}).$$

**ЧИСЛЕННОЕ РЕШЕНИЕ
ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ**

**§1. ЧИСЛЕННОЕ РЕШЕНИЕ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ
ПЕРВОГО ПОРЯДКА**

1⁰. Понятие о численном решении задачи Коши

Дифференциальное уравнение первого порядка, разрешенное относительно производной, имеет вид

$$y' = f(x, y). \quad (1)$$

Решением дифференциального уравнения (1) называется функция $\varphi(x)$, подстановка которой в уравнение обращает его в тождество: $\varphi'(x) = f(x, \varphi(x))$. График решения $y = \varphi(x)$ называется интегральной кривой. Например, решением уравнения $y' = y$ является функция $\varphi(x) = Ce^x$ при любом значении произвольной постоянной C .

Задача Коши для дифференциального уравнения (1) состоит в том, чтобы найти решение уравнения (1), удовлетворяющее начальному условию

$$y|_{x=x_0} = y_0. \quad (2)$$

Пару чисел (x_0, y_0) называют начальными данными. Решение задачи Коши называется частным решением уравнения (1) при условии (2). Например, частным решением задачи Коши

$$y' = y, \quad y|_{x=0} = 1$$

является функция $\varphi(x) = e^x$.

Частному решению соответствует одна из интегральных кривых, проходящая через точку (x_0, y_0) (рис.23).

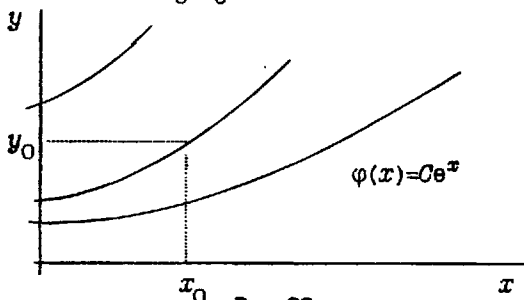


Рис.23

Условия существования и единственности решения задачи Коши содержатся в следующей теореме.

Теорема 1. Пусть функция $f(x, y)$ – правая часть дифференциального уравнения (1) – непрерывна вместе со своей частной производной

$\frac{\partial f(x, y)}{\partial y}$ по переменной y в некоторой области D на плоскости.

Тогда при любых начальных данных $(x_0, y_0) \in D$ задача Коши (1)–(2) имеет единственное решение $y = \varphi(x)$.

При выполнении условий теоремы через точку (x_0, y_0) на плоскости проходит единственная интегральная кривая. Будем считать, что условия теоремы существования и единственности выполняются.

Численное решение задачи Коши (1)–(2) состоит в том, чтобы получить искомое решение $\varphi(x)$ в виде таблицы его приближенных значений для заданных значений аргумента x на некотором отрезке $[a, b]$:

$$x_0 = a, x_1, x_2, \dots, x_m = b. \quad (3)$$

Точки (3) называют **узловыми точками**, а множество этих точек называют **сеткой** на отрезке $[a, b]$. Будем использовать **равномерную сетку** с шагом h :

$$h = (b-a)/m; \quad x_i - x_{i-1} = h \text{ или } x_i = x_0 + ih \quad (i=1, \dots, m).$$

Приближенные значения численного решения задачи Коши в узловых точках x_i обозначим через y_i ; таким образом,

$$y_i \approx \varphi(x_i) \quad (i=1, 2, \dots, m).$$

Для любого численного метода решения задачи (1)–(2) начальное условие (2) выполняется точно, т.е. $y_0 = \varphi(x_0)$.

Величина погрешности численного метода решения задачи Коши на сетке отрезка $[a, b]$ оценивается величиной

$$d = \max_{1 \leq i \leq m} \{ |y_i - \varphi(x_i)| \},$$

т.е. расстоянием между векторами приближенного решения $(y_0, y_1, \dots, y_m)$ и точного решения $(\varphi(x_0), \varphi(x_1), \dots, \varphi(x_m))$ на сетке по m -норме. Говорят, что численный метод имеет p -**й порядок точности** по шагу h на сетке, если расстояние d можно представить в виде степенной функции от h :

$$d = Ch^p, \quad p > 0,$$

где C – некоторая положительная постоянная, зависящая от правой

Оценим погрешность метода Эйлера на одном шаге. Для этого запишем разложение точного решения задачи Коши в точке x_1 по формуле Тейлора с остаточным членом в форме Лагранжа:

$$\begin{aligned}\varphi(x_1) &= \varphi(x_0 + h) = \varphi(x_0) + \varphi'(x_0)h + \frac{1}{2}\varphi''(\xi)h^2 = \\ &= y_0 + hf(x_0, y_0) + \frac{1}{2}\varphi''(\xi)h^2 = y_1 + \frac{1}{2}\varphi''(\xi)h^2, \quad \xi \in (x_0, x_1).\end{aligned}$$

Погрешность метода на одном шаге имеет порядок h^2 , так как

$$d_0 = |\varphi(x_1) - y_1| = \frac{1}{2}|\varphi''(\xi)| \cdot h^2 \leq \frac{1}{2} \max_{a \leq x \leq b} |\varphi''(x)| \cdot h^2$$

После m шагов погрешность вычисления значения y_m в конечной точке отрезка возрастет не более чем в m раз. Погрешность метода Эйлера можно оценить неравенством

$$d \leq \frac{1}{2} \max_{a \leq x \leq b} |\varphi''(x)| \cdot h^2 m = \frac{1}{2} \max_{a \leq x \leq b} |\varphi''(x)| \cdot (b-a)h$$

или представить в виде $d = Ch$, где $C \in \left[0, \frac{1}{2} \max_{a \leq x \leq b} |\varphi''(x)| \cdot (b-a)\right]$.

Это означает, что метод Эйлера имеет первый порядок точности. В частности, при уменьшении шага h в 10 раз погрешность уменьшится примерно в 10 раз.

Практическую оценку погрешности решения, найденного на сетке с шагом $h/2$, в точке $x_i \in [a, b]$ производят с помощью приближенного равенства - правила Рунге:

$$|\varphi(x_i) - y_i(h/2)| \sim \frac{|y_i(h) - y_i(h/2)|}{2^p - 1}, \quad (5)$$

где p - порядок точности численного метода. Таким образом, оценка полученного результата по формуле (5) вынуждает проводить вычисления дважды: один раз с шагом h , другой - с шагом $h/2$.

Пример 1. Решить задачу Коши

$$y' = x + y, \quad y|_{x=0} = 1$$

методом Эйлера на отрезке $[0; 0.4]$. Найти решение на равномерной сетке с шагом $h = 0.1$ в четырех узловых точках. С помощью программы Приложения (см. с. 313-317) найти решение в тех же узлах, ведя расчет с уменьшенным вдвое шагом. Вычислить пог-

решности приближений при расчете с шагом $h=0,05$: а) с помощью формулы (5); б) сравним с точным значением. Аналитическое решение задачи имеет вид $\varphi(x) = 2e^x - x - 1$.

Решение. Здесь $f(x, y) = x + y$; $m = 4$; $a = 0$; $b = 0,4$; $h = (b - a)/m = 0,4/4 = 0,1$. Используя рекуррентные формулы

$$x_0=0; y_0=1; \quad x_i=x_{i-1}+0,1; \quad y_i=y_{i-1}+0,1(x_{i-1}+y_{i-1}) \quad (i=1,2,3,4),$$

последовательно находим

$$\text{при } i = 1: \quad x_1 = 0,1; \quad y_1 = 1 + 0,1(0 + 1) = 1,1;$$

$$\text{при } i = 2: \quad x_2 = 0,2; \quad y_2 = 1,1 + 0,1(0,1 + 1,1) = 1,22;$$

$$\text{при } i = 3: \quad x_3 = 0,3; \quad y_3 = 1,22 + 0,1(0,2 + 1,22) = 1,362;$$

$$\text{при } i = 4: \quad x_4 = 0,4; \quad y_4 = 1,362 + 0,1(0,3 + 1,362) = 1,5282.$$

Обозначим $\tilde{d}_i = |y_i(h) - y_i(h/2)|$, $d_i = |\varphi(x_i) - y_i(h/2)|$ и представим результаты вычислений в таблице

i	x_i	$y_i(h)$	$y_i(h/2)$	$\varphi(x_i)$	$\tilde{d}_i$	d_i
1	0,1	1,1	1,105	1,110342	0,005	0,005342
2	0,2	1,22	1,231012	1,242805	0,011012	0,011793
3	0,3	1,362	1,380191	1,399718	0,018191	0,019527
4	0,4	1,5282	1,554911	1,583649	0,026711	0,028738

Отметим, что оценки погрешностей $\tilde{d}_i$ решения $y_i(h/2)$, вычисляемых по формулам (5), близки к отклонениям d_i и обе величины достигают значения $d \approx 0,03$ — ошибки метода Эйлера при вычислении с шагом 0,05. Для сравнения заметим, что погрешность при вычислениях с шагом 0,1 составляет $d = |y_4 - \varphi(x_4)| \approx 0,06$. ■

3°. Методы Рунге — Кутты

Численные методы решения задачи Коши

$$y' = f(x, y), \quad y|_{x=x_0} = y_0$$

на равномерной сетке $\{x_0=a, x_1, x_2, \dots, x_m=b\}$ отрезка $[a, b]$ с шагом

$h=(b-a)/m$ являются методами Рунге - Кутта, если, начиная с данных (x_0, y_0) , решение ведется по следующим рекуррентным формулам:

$$x_t = x_{t-1} + h, \quad y_t = y_{t-1} + \Delta y_{t-1} \quad (t=1, 2, \dots, m),$$

$$\Delta y_{t-1} = \sum_{j=1}^p d_j k_j^{[t-1]}, \quad k_j^{[t-1]} = h f(x_{t-1} + c_j h, y_{t-1} + c_j k_{j-1}^{[t-1]}). \quad (6)$$

Метод называют методом Рунге - Кутта порядка p , если он имеет p -й порядок точности по шагу h на сетке. Порядок точности p достигается с помощью формул (6) при определенных значениях коэффициентов c_j и d_j ($j=1, 2, \dots, p$); c_1 всегда полагают равным нулю.

Эти коэффициенты вычисляют по следующей схеме:

1) точное решение $\varphi(x_0+h)$ и его приближение $y_1 = y_0 + \Delta y_0(h)$ представляют в виде разложения по формуле Тейлора с центром x_0 вплоть до слагаемого порядка h^{p+1} ;

2) из равенств подобных членов при одинаковых степенях h в двух разложениях получают уравнения, решая которые находят коэффициенты c_j и d_j .

Заметим, что метод Эйлера можно называть методом Рунге - Кутта первого порядка. Действительно, для $p=1$, $c_1=0$, $d_1=1$ формулы (6) преобразуются в соотношения (4):

$$x_t = x_{t-1} + h, \quad y_t = y_{t-1} + \Delta y_{t-1} \quad (t=1, 2, \dots, m),$$

$$\Delta y_{t-1} = d_1 k_1^{[t-1]} = k_1^{[t-1]}, \quad k_1^{[t-1]} = h f(x_{t-1}, y_{t-1}) \quad \text{или}$$

$$x_t = x_{t-1} + h, \quad y_t = y_{t-1} + h f(x_{t-1}, y_{t-1}).$$

Метод Рунге - Кутта второго порядка называют методом Эйлера - Коши, если $p=2$, $c_1=0$, $c_2=1$, $d_1=d_2=\frac{1}{2}$. Алгоритм метода Эйлера - Коши получается из формул (6):

$$x_t = x_{t-1} + h, \quad y_t = y_{t-1} + \Delta y_{t-1}, \quad \Delta y_{t-1} = \frac{1}{2} \cdot [k_1^{[t-1]} + k_2^{[t-1]}] \quad (t=1, \dots, m), \quad (7)$$

$$k_1^{[t-1]} = h f(x_{t-1}, y_{t-1}), \quad k_2^{[t-1]} = h f(x_{t-1} + h, y_{t-1} + h f(x_{t-1}, y_{t-1})).$$

Для практической оценки погрешности решения можно применять пра-

вию Рунге, полагая в формуле (5) $p = 2$.

Пример 2. Решить задачу Коши

$$y' = x + y, \quad y|_{x=0} = 1$$

методом Эйлера - Коши на отрезке $[0; 0,4]$. Найти решение на равномерной сетке с шагом $0,1$ в четырех узловых точках.

Решение. Формулы (7) в данном случае примут вид

$$k_1^{[t-1]} = h(x_{t-1} + y_{t-1}), \quad k_2^{[t-1]} = h(x_{t-1} + h + y_{t-1} + k_1^{[t-1]}),$$

$$x_t = x_{t-1} + h, \quad y_t = y_{t-1} + \frac{1}{2} [k_1^{[t-1]} + k_2^{[t-1]}] \quad (t=1, 2, 3, 4).$$

Полагая $x_0 = 0$, $y_0 = 1$, последовательно находим

при $t = 1$:

$$k_1^{[0]} = 0,1(0+1) = 0,1; \quad k_2^{[0]} = 0,1(0+0,1+1+0,1) = 0,12;$$

$$x_1 = 0+0,1 = 0,1; \quad y_1 = 1 + \frac{1}{2}(0,1+0,12) = 1,11;$$

при $t = 2$:

$$k_1^{[1]} = 0,1(0,1+1,11) = 0,121; \quad k_2^{[1]} = 0,1(0,1+0,1+1,11+0,121) = 0,1431;$$

$$x_2 = 0,1+0,1 = 0,2; \quad y_2 = 1,11 + \frac{1}{2}(0,121+0,1431) = 1,24205.$$

Далее получаем при $t = 3$: $x_3 = 0,3$; $y_3 = 1,398465$;

при $t = 4$: $x_4 = 0,4$; $y_4 = 1,581804$.

Погрешность полученного решения не превышает величины $|y_4 - \varphi(x_4)| \approx 0,002$. ■

Метод Рунге - Кутта четвертого порядка называют **классическим методом Рунге - Кутта**, если $p = 4$, $c_1 = 0$, $c_2 = c_3 = \frac{1}{2}$, $c_4 = 1$, $a_1 = a_4 = \frac{1}{6}$, $a_2 = a_3 = \frac{1}{3}$. Из рекуррентных формул (6) получим алгоритм решения задачи Коши классическим методом Рунге - Кутта:

$$x_i = x_{i-1} + h, \quad y_i = y_{i-1} + \Delta y_{i-1} \quad (i=1, 2, \dots, m),$$

$$\Delta y_{i-1} = \frac{1}{6} \left[k_1^{[i-1]} + 2k_2^{[i-1]} + 2k_3^{[i-1]} + k_4^{[i-1]} \right],$$

$$k_1^{[i-1]} = h f(x_{i-1}, y_{i-1}),$$

$$k_2^{[i-1]} = h f\left(x_{i-1} + \frac{1}{2}h, y_{i-1} + \frac{1}{2}k_1^{[i-1]}\right), \quad (8)$$

$$k_3^{[i-1]} = h f\left(x_{i-1} + \frac{1}{2}h, y_{i-1} + \frac{1}{2}k_2^{[i-1]}\right),$$

$$k_4^{[i-1]} = h f\left(x_{i-1} + h, y_{i-1} + k_3^{[i-1]}\right).$$

Графиком приближенного решения является ломаная, последовательно соединяющая точки $P_i(x_i, y_i)$ ($i=0, 1, 2, \dots, m$). С увеличением порядка численного метода звенья ломаной приближаются к ломаной, образованной хордами интегральной кривой $y=\varphi(x)$, последовательно соединяющими точки $(x_i, \varphi(x_i))$ на интегральной кривой.

Правило Рунге (5) практической оценки погрешности решения для численного метода четвертого порядка имеет вид

$$|\varphi(x_i) - y_i(h/2)| \approx \frac{1}{15} |y_i(h) - y_i(h/2)|.$$

Пример 3. Решить задачу Коши

$$y' = x + y, \quad y|_{x=0} = 1$$

классическим методом Рунге - Кутты на отрезке $[0; 0,4]$. Найти решение на равномерной сетке с шагом 0,1 в четырех узловых точках.

Решение. Так как $f(x, y) = x + y$, то согласно формулам (8) получаем

$$k_1^{[i-1]} = h(x_{i-1} + y_{i-1}), \quad k_2^{[i-1]} = h\left(x_{i-1} + \frac{1}{2}h + y_{i-1} + \frac{1}{2}k_1^{[i-1]}\right),$$

$$k_3^{[i-1]} = h\left(x_{i-1} + \frac{1}{2}h + y_{i-1} + \frac{1}{2}k_2^{[i-1]}\right), \quad k_4^{[i-1]} = h(x_{i-1} + h + y_{i-1} + k_3^{[i-1]}),$$

$$x_i = x_{i-1} + h, \quad y_i = y_{i-1} + \frac{1}{6} \left[k_1^{[i-1]} + 2k_2^{[i-1]} + 2k_3^{[i-1]} + k_4^{[i-1]} \right]$$

для значений $i = 1, 2, 3, 4$.

Полагая $x_0 = 0$, $y_0 = 1$, последовательно находим

при $t = 1$:

$$k_1^{[0]} = 0,1(0 + 1) = 0,1; \quad k_2^{[0]} = 0,1(0 + 0,05 + 1 + 0,05) = 0,11;$$

$$k_3^{[0]} = 0,1(0 + 0,05 + 1 + 0,055) = 0,1105;$$

$$k_4^{[0]} = 0,1(0 + 0,1 + 1 + 0,1105) = 0,121050;$$

$$x_1 = 0 + 0,1 = 0,1; \quad y_1 = 1 + \frac{1}{6}(0,1 + 2 \cdot 0,11 + 2 \cdot 0,1105 + 0,12105) = 1,110342;$$

при $t = 2$:

$$k_1^{[1]} = 0,1(0,1 + 1,110342) = 0,1210342;$$

$$k_2^{[1]} = 0,1(0,1 + 0,05 + 1,110342 + 0,0605171) = 0,1320859;$$

$$k_3^{[1]} = 0,1(0,1 + 0,05 + 1,110342 + 0,06604295) = 0,1326385;$$

$$k_4^{[1]} = 0,1(0,1 + 0,1 + 1,110342 + 0,1326385) = 0,1442980;$$

$$x_2 = 0,1 + 0,1 = 0,2; \quad y_2 = y_1 + \frac{1}{6} \left(k_1^{[1]} + 2k_2^{[1]} + 2k_3^{[1]} + k_4^{[1]} \right) = 1,242805.$$

Далее получаем при $t = 3$: $x_3 = 0,3$; $y_3 = 1,399717$;

при $t = 4$: $x_4 = 0,4$; $y_4 = 1,583648$.

Погрешность полученного решения не превышает величины $|y_4 - \varphi(x_4)| \sim 0,000001$. ■

Для наглядности численные решения одной и той же задачи Коши, рассмотренные в примерах 1–3, сведены в одну таблицу:

t	x_t	Значения y_t , найденные методом			Точное решение $\varphi(x_t) = 2e^{x_t} - x_t - 1$
		Эйлера	Эйлера - Коши	Рунге - Кутты	
0	0,0	1,0	1,0	1,0	1,0
1	0,1	1,1	1,11	1,110342	1,110342
2	0,2	1,22	1,24205	1,242805	1,242805
3	0,3	1,362	1,398465	1,399717	1,399718
4	0,4	1,5282	1,581804	1,583648	1,583649

Упражнения

Найти решение задачи Коши для дифференциального уравнения первого порядка на равномерной сетке отрезка $[a, b]$ один раз с шагом $h=0,2$, другой - с шагом $0,1$ методами Эйлера, Эйлера - Коши и классическим методом Рунге - Кутты. Оценить погрешность численного решения по принципу Рунге. Сравнить численное решение с точным. Результаты представить в виде таблиц, аналогичных приведенным в примерах этого параграфа.

В задачах 16-25 либо начальные данные, либо значения концов отрезка $[a, b]$ представлены иррациональными числами. При решении этих задач в программах из Приложения к данному параграфу лучше заменить операторы ввода данных с клавиатуры. Например, если в условии задан отрезок интегрирования, равный $[e, e+1]$, то в программе на языке BASIC оператор "INPUT a, b" следует заменить на операторы "a = EXP(1): b = a + 1".

1. $y' = \frac{1+xy}{x^2}$, $y|_{x=1} = 0$, $1 \leq x \leq 2$, $\varphi(x) = \frac{1}{2} \left(x - \frac{1}{x} \right)$.
2. $y' = y - \frac{2x}{y}$, $y|_{x=0} = 1$, $0 \leq x \leq 1$, $\varphi(x) = \sqrt{2x+1}$.
3. $y' = x + \frac{3y}{x}$, $y|_{x=1} = 0$, $1 \leq x \leq 2$, $\varphi(x) = x^2(x-1)$.
4. $y' = xy$, $y|_{x=0} = 1$, $0 \leq x \leq 1$, $\varphi(x) = e^{x^2/2}$.
5. $y' = \frac{y^2+xy}{x^2}$, $y|_{x=1} = 1$, $1 \leq x \leq 2$, $\varphi(x) = x/(1-\ln x)$.
6. $y' = \frac{1-y+\ln x}{x}$, $y|_{x=1} = 0$, $1 \leq x \leq 2$, $\varphi(x) = \ln x$.
7. $y' = \frac{x+y}{x}$, $y|_{x=1} = 0$, $1 \leq x \leq 2$, $\varphi(x) = x \ln x$.
8. $y' + 2xy = xe^{-x^2}$, $y|_{x=0} = 0$, $0 \leq x \leq 1$, $\varphi(x) = \frac{1}{2} x^2 e^{-x^2}$.
9. $y' + y \cos x = \frac{1}{2} \sin 2x$, $y|_{x=0} = 0$, $0 \leq x \leq 1$, $\varphi(x) = \sin x + e^{-\sin x} - 1$.
10. $y' + y \tan x = \sin 2x$, $y|_{x=0} = -1$, $0 \leq x \leq 1$, $\varphi(x) = (1-2\cos x) \cos x$.
11. $xy' - y^2 \ln x + y = 0$, $y|_{x=1} = 1$, $1 \leq x \leq 2$, $\varphi(x) = 1/(1+\ln x)$.

$$12. xy' = x + y + x \exp\left(\frac{y}{x}\right), \quad y|_{x=1} = 0, \quad 1 \leq x \leq 1,9, \quad \varphi(x) = x \ln \frac{x}{2-x}.$$

$$13. x^2 y' - y = x^2 \exp(x-1/x), \quad y|_{x=1} = 1, \quad 1 \leq x \leq 2, \quad \varphi(x) = \exp((x^2-1)/x).$$

$$14. (x^2+1)y' + xy - 1 = 0, \quad y|_{x=0} = 0, \quad 0 \leq x \leq 1, \quad \varphi(x) = \frac{\ln(x + \sqrt{x^2+1})}{\sqrt{x^2+1}}.$$

$$15. xy' - y = \frac{x}{\ln x}, \quad y|_{x=e} = 0, \quad e \leq x \leq e+1, \quad \varphi(x) = x \ln \ln x.$$

$$16. xy' - y = x^2 \sin x, \quad y|_{x=\frac{\pi}{2}} = 0, \quad \frac{\pi}{2} \leq x \leq \frac{\pi}{2} + 1, \quad \varphi(x) = -x \cos x.$$

$$17. xy' + y = x \sin x, \quad y|_{x=\frac{\pi}{2}} = 0, \quad \frac{\pi}{2} \leq x \leq \frac{\pi}{2} + 1, \quad \varphi(x) = \frac{\sin x - 1}{x} - \cos x.$$

$$18. y' + e^{x-y} = e^{x(1-x)} + 2x, \quad y|_{x=0} = 0, \quad 0 \leq x \leq 1, \quad \varphi(x) = x^2.$$

$$19. xy' = y \ln y, \quad y|_{x=1} = e, \quad 1 \leq x \leq 2, \quad \varphi(x) = e^x.$$

$$20. xy' - y(\ln(xy)-1) = 0, \quad y|_{x=1} = e, \quad 1 \leq x \leq 2, \quad \varphi(x) = e^x/x.$$

$$21. xy' = x \sin \frac{y}{x} + y, \quad y|_{x=1} = \frac{\pi}{2}, \quad 1 \leq x \leq 2, \quad \varphi(x) = 2x \arctg x.$$

$$22. x^2 y' = (x-1)y, \quad y|_{x=1} = e, \quad 1 \leq x \leq 2, \quad \varphi(x) = x \exp(1/x).$$

$$23. (x^2+1)y' + xy = x(x^2+1), \quad y|_{x=\sqrt{2}} = 1, \quad \sqrt{2} \leq x \leq \sqrt{2}+1, \quad \varphi(x) = \frac{x^2+1}{3}.$$

$$24. y' - \frac{y}{x \ln x} = x \ln x, \quad y|_{x=e} = \frac{e^2}{2}, \quad e \leq x \leq e+1, \quad \varphi(x) = \frac{x^2}{2} \ln x.$$

$$25. y' - y \operatorname{ctg} x = \sin x, \quad y|_{x=\frac{\pi}{2}} = 0, \quad \frac{\pi}{2} \leq x \leq \frac{\pi}{2} + 1, \quad \varphi(x) = \left(x - \frac{\pi}{2}\right) \sin x.$$

§2. ЧИСЛЕННОЕ РЕШЕНИЕ СИСТЕМ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

ПЕРВОГО ПОРЯДКА

Пусть дана система двух дифференциальных уравнений первого порядка:

$$\begin{cases} y_1' = f_1(x, y_1, y_2), \\ y_2' = f_2(x, y_1, y_2). \end{cases} \quad (9)$$

Решением системы (9) называется пара функций $\varphi_1(x)$ и $\varphi_2(x)$, при подстановке которых в систему получаются тождества:

$$\varphi_1'(x) = f_1(x, \varphi_1(x), \varphi_2(x)), \quad \varphi_2'(x) = f_2(x, \varphi_1(x), \varphi_2(x)).$$

Решению

$$\begin{cases} y_1 = \varphi_1(x), \\ y_2 = \varphi_2(x) \end{cases}$$

системы уравнений (9) соответствует интегральная кривая в пространстве трех измерений (x, y_1, y_2) (рис.25). Условия, при которых через каждую точку $P_0(x_0, y_{10}, y_{20})$ некоторой области D трехмерного пространства проходит единственная интегральная кривая, содержится в теореме существования и единственности решения.

Теорема 2. Если функции $f_1(x, y_1, y_2)$ и $f_2(x, y_1, y_2)$ — правые части дифференциальных уравнений системы (9) — непрерывны вместе со своими частными производными по переменным y_1 и y_2 в некоторой области D трехмерного пространства, то для любой точки $(x_0, y_{10}, y_{20}) \in D$ система (9) имеет единственное решение, удовлетворяющее начальным условиям

$$y_1|_{x=x_0} = y_{10}, \quad y_2|_{x=x_0} = y_{20}. \quad (10)$$

Задача Коши для системы состоит в нахождении решения системы (9), удовлетворяющего начальным условиям (10).

Постановка задачи Коши для системы n дифференциальных уравнений первого порядка аналогична задаче (9)–(10), а именно: требуется найти решение системы

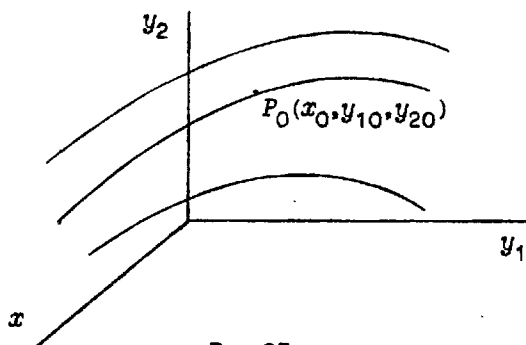


Рис.25

$$\begin{cases} y_1' = f_1(x, y_1, y_2, \dots, y_n), \\ y_2' = f_2(x, y_1, y_2, \dots, y_n), \\ \vdots \\ y_n' = f_n(x, y_1, y_2, \dots, y_n) \end{cases} \quad (11)$$

ПРИ НАЧАЛЬНЫХ УСЛОВИЯХ

$$y_1|_{x=x_0} = y_{10}, \quad y_2|_{x=x_0} = y_{20}, \dots, y_n|_{x=x_0} = y_{n0}. \quad (12)$$

Теорема существования и единственности решения задачи Коши (11)-(12) имеет формулировку, аналогичную приведенной для частного случая $n = 2$.

Если ввести векторные обозначения

$$\mathbf{y}(x) = \begin{bmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_n(x) \end{bmatrix}, \quad \mathbf{y}'(x) = \begin{bmatrix} y'_1(x) \\ y'_2(x) \\ \vdots \\ y'_n(x) \end{bmatrix}, \quad \mathbf{f}(x, \mathbf{y}) = \begin{bmatrix} f_1(x, \mathbf{y}) \\ f_2(x, \mathbf{y}) \\ \vdots \\ f_n(x, \mathbf{y}) \end{bmatrix}, \quad \mathbf{y}_0 = \begin{bmatrix} y_{10} \\ y_{20} \\ \vdots \\ y_{n0} \end{bmatrix}.$$

то задача Коши (11)–(12) в векторной форме запишется так:

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \quad \mathbf{y}|_{x=x_0} = \mathbf{y}_0. \quad (13)$$

Численное решение задачи Коши (13) состоит в том, что на сетке отрезка $[a, b]$ требуется получить приближенные значения координат вектора $y(x)$ в узлах сетки x_i ($i=1, 2, \dots, m$).

Обозначим вектор, аппроксимирующий решение, через $y_i \approx y(x_i)$ ($i = 1, 2, \dots, m$), а его координаты - через y_{ki} ($k = 1, 2, \dots, n$; $i = 1, 2, \dots, m$) так, что $y_{ki} \approx y_k(x_i)$ или

$$\mathbf{y}_t = \begin{bmatrix} y_{1t} \\ y_{2t} \\ \vdots \\ y_{nt} \end{bmatrix} \approx \mathbf{y}(x_t) = \begin{bmatrix} y_1(x_t) \\ y_2(x_t) \\ \vdots \\ y_n(x_t) \end{bmatrix} \quad (t=1, \dots, m).$$

Будем искать решение на равномерной сетке с шагом $h = (b-a)/m$.

Величина погрешности численного метода оценивается величиной $d = \max_{1 \leq i \leq m} \{d_i\}$, где d_i - погрешность решения на сетке с шагом h

В точке x_0 :

$$d_i(h) = \max_{1 \leq k \leq n} \{ |y_{ki}(h) - y_k(x_i)| \}.$$

Практически погрешность в точке x_i оценивается по формуле Рунге, аналогичной (5). А именно, пусть

$$y_i(h) = \begin{bmatrix} y_{1i}(h) \\ y_{2i}(h) \\ \vdots \\ y_{ni}(h) \end{bmatrix}, \quad y_i(h/2) = \begin{bmatrix} y_{1i}(h/2) \\ y_{2i}(h/2) \\ \vdots \\ y_{ni}(h/2) \end{bmatrix}$$

— значения численного решения в точке x_i , полученные для шагов h и $h/2$ соответственно; тогда погрешность d_i в точке x_i для вычислений с шагом $h/2$ выражается приближенным равенством

$$d_i(h/2) \approx \frac{\max_{1 \leq k \leq n} \{|y_{ki}(h) - y_{ki}(h/2)|\}}{2^p - 1}, \quad (14)$$

где p — порядок точности численного метода.

Численное решение задачи Коши для системы дифференциальных уравнений находится с помощью классического метода Рунге — Кутты четвертого порядка. Векторная форма алгоритма метода Рунге — Кутты для задачи (13) соответствует рекуррентным формулам (8) и имеет вид

$$x_t = x_{t-1} + h, \quad y_t = y_{t-1} + \Delta y_{t-1} \quad (t=1, 2, \dots, m),$$

$$\Delta y_{t-1} = \frac{1}{6} \left[k_1^{[t-1]} + 2k_2^{[t-1]} + 2k_3^{[t-1]} + k_4^{[t-1]} \right],$$

$$k_1^{[t-1]} = hf(x_{t-1}, y_{t-1}),$$

$$k_2^{[t-1]} = hf\left(x_{t-1} + \frac{1}{2}h, y_{t-1} + \frac{1}{2}k_1^{[t-1]}\right), \quad (15)$$

$$k_3^{[t-1]} = hf\left(x_{t-1} + \frac{1}{2}h, y_{t-1} + \frac{1}{2}k_2^{[t-1]}\right),$$

$$k_4^{[t-1]} = hf(x_{t-1} + h, y_{t-1} + k_3^{[t-1]}),$$

где векторы $k_j^{[t-1]} = \begin{bmatrix} k_{j1}^{[t-1]} \\ k_{j2}^{[t-1]} \\ \vdots \\ k_{jn}^{[t-1]} \end{bmatrix} \quad (j=1, 2, 3, 4).$

Пример. Найти численное решение задачи Коши для системы двух дифференциальных уравнений

$$\begin{cases} y_1' = y_2, \\ y_2' = -y_1, \end{cases} \quad y_1|_{x=0} = 0, \quad y_2|_{x=0} = 1$$

на сетке отрезка $\left[0, \frac{\pi}{2}\right]$ методом Рунге - Кутты. Вычисления провести с шагами $\frac{\pi}{6}$ и $\frac{\pi}{12}$. Оценить погрешность по принципу Рунге. Сравнить численное решение с аналитическим решением $y_1 = \sin x$, $y_2 = \cos x$.

Решение. Здесь

$$f_1(x, y_1, y_2) = y_2, \quad f_2(x, y_1, y_2) = -y_1,$$

$$x_0 = 0, \quad y_{10} = 0, \quad y_{20} = 1, \quad h = \pi/6 = 0,523599.$$

Численное решение будем искать по формулам (15).

Последовательно вычисляя, при $t=1$ и $j=1, 2, 3, 4$ имеем:

$$k_1^{[0]} = \begin{bmatrix} k_{11}^{[0]} \\ k_{12}^{[0]} \end{bmatrix}; \quad k_{11}^{[0]} = hf_1(x_0, y_{10}, y_{20}) = hy_{20} = h \cdot 1 = 0,523599;$$

$$k_{12}^{[0]} = hf_2(x_0, y_{10}, y_{20}) = -hy_{10} = -h \cdot 0 = 0;$$

$$k_2^{[0]} = \begin{bmatrix} k_{21}^{[0]} \\ k_{22}^{[0]} \end{bmatrix}; \quad k_{21}^{[0]} = hf_1\left(x_0 + \frac{1}{2}h, y_{10} + \frac{1}{2}k_{11}^{[0]}, y_{20} + \frac{1}{2}k_{12}^{[0]}\right) =$$

$$= h(y_{20} + \frac{1}{2}k_{12}^{[0]}) = 0,523599 (1 + 0) = 0,523599;$$

$$k_{22}^{[0]} = hf_2\left(x_0 + \frac{1}{2}h, y_{10} + \frac{1}{2}k_{11}^{[0]}, y_{20} + \frac{1}{2}k_{12}^{[0]}\right) =$$

$$= -h(y_{10} + \frac{1}{2}k_{11}^{[0]}) = -0,523599 \cdot 0,261799 = -0,137078;$$

$$k_3^{[0]} = \begin{bmatrix} k_{31}^{[0]} \\ k_{32}^{[0]} \end{bmatrix}; \quad k_{31}^{[0]} = hf_1\left(x_0 + \frac{1}{2}h, y_{10} + \frac{1}{2}k_{21}^{[0]}, y_{20} + \frac{1}{2}k_{22}^{[0]}\right) =$$

$$= h(y_{20} + \frac{1}{2}k_{22}^{[0]}) = 0,523599 (1 - 0,068539) = 0,487712;$$

$$k_{32}^{[0]} = hf_2\left(x_0 + \frac{1}{2}h, y_{10} + \frac{1}{2}k_{21}^{[0]}, y_{20} + \frac{1}{2}k_{22}^{[0]}\right) =$$

$$= -h(y_{10} + \frac{1}{2}k_{21}^{[0]}) = -0,523599 \cdot 0,261799 = -0,137078;$$

$$k_4^{[0]} = \begin{bmatrix} k_{41}^{[0]} \\ k_{42}^{[0]} \end{bmatrix}; \quad k_{41}^{[0]} = hf_1(x_0 + h, y_{10} + k_{31}^{[0]}, y_{20} + k_{32}^{[0]}) =$$

$$= h(y_{20} + k_{32}^{[0]}) = 0,523599 (1 - 0,137078) = 0,451825;$$

$$k_{42}^{[0]} = hf_2(x_0 + h, y_{10} + k_{31}^{[0]}, y_{20} + k_{32}^{[0]}) =$$

$$= -h(y_{10} + k_{31}^{[0]}) = -0,523599 \cdot 0,487712 = -0,255366;$$

$$y_1 = \begin{bmatrix} y_{11} \\ y_{21} \end{bmatrix}; \quad y_{11} = y_{10} + \frac{1}{6} (k_{11}^{[0]} + 2k_{21}^{[0]} + 2k_{31}^{[0]} + k_{41}^{[0]}) =$$

$$= (0,523599 + 2 \cdot 0,523599 + 2 \cdot 0,487712 + 0,451825) / 6 =$$

$$= 0,499674;$$

$$y_{21} = y_{10} + \frac{1}{6} (k_{12}^{[0]} + 2k_{22}^{[0]} + 2k_{32}^{[0]} + k_{42}^{[0]}) =$$

$$= 1 + (0 + 2(-0,137078) + 2(-0,137078) - 0,255366) / 6 =$$

$$= 0,866054;$$

$$x_1 = x_0 + h = 0,523599.$$

Продолжая процесс вычислений, получаем

$$\text{при } t=2: \quad x_2 = 1,047197; \quad y_{12} = 0,865489; \quad y_{22} = 0,500375;$$

$$\text{при } t=3: \quad x_3 = 1,570796; \quad y_{13} = 0,999585; \quad y_{23} = 0,000889.$$

Результаты численного решения задачи с шагами $\pi/6$ и $\pi/12$ сведены в таблицу

t	x_t	Численное решение задачи Коши				Точное решение $y_1 = \sin x_t$
		с шагом $\pi/6$		с шагом $\pi/12$		
		y_{1t}	y_{2t}	y_{1t}	y_{2t}	
0	0	0	1	0	1	0
1	$\pi/6=0,523599$	0,499674	0,866054	0,499980	0,866032	0,5
2	$\pi/3=1,047197$	0,865489	0,500375	0,865998	0,500030	0,866253
3	$\pi/2=1,570796$	0,999585	0,000889	0,999987	0,000060	1

Используя правило Рунге, находим погрешность

$$\frac{\max_{1 \leq i \leq 15} \{ \max_{1 \leq k \leq 2} |y_{hi}(h) - y_{hi}(h/2)| \}}{15} =$$

$$= \frac{|y_{23}(h) - y_{23}(h/2)|}{15} = \frac{0,00829}{15} = 0,000055. \blacksquare$$

§3. ЧИСЛЕННОЕ РЕШЕНИЕ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ И СИСТЕМ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ ВЫСШИХ ПОРЯДКОВ

Задача Коши для дифференциального уравнения n -го порядка ставится так: найти решение уравнения

$$y^{(n)} = f(x, y, y', y'', \dots, y^{(n-1)}) \quad (16)$$

при начальных условиях

$$y|_{x=x_0} = y_0, \quad y'|_{x=x_0} = y'_0, \quad \dots, \quad y^{(n-1)}|_{x=x_0} = y_0^{(n-1)}. \quad (17)$$

Задача Коши (16)–(17) для дифференциального уравнения n -го порядка приводится к задаче Коши для системы n дифференциальных уравнений первого порядка (11)–(12), к которой затем применяют численные методы решения систем.

Положим

$$y_1 = y, \quad y_2 = y', \quad y_3 = y'', \quad \dots, \quad y_n = y_{n-1}'$$

и выразим функцию $y(x)$ вместе с ее производными до $(n-1)$ -го порядка включительно через введенные функции:

$$y = y_1, \quad y' = y_2, \quad y'' = y_3, \quad \dots, \quad y^{(n-1)} = y_n.$$

Вместо задачи (16)–(17) имеем задачу для системы уравнений

$$\begin{cases} y_1' = y_2, \\ y_2' = y_3, \\ \dots \\ y_{n-1}' = y_n, \\ y_n' = f_n(x, y_1, y_2, \dots, y_n) \end{cases} \quad (18)$$

при начальных условиях

$$y_1|_{x=x_0} = y_0, \quad y_2|_{x=x_0} = y'_0, \quad \dots, \quad y_n|_{x=x_0} = y_0^{(n-1)}. \quad (19)$$

Подчеркнем, что численное решение задачи Коши (16)–(17) – это таблица значений функции $y_i = y_1$ в точках x_i ($i=1, 2, \dots, n$).

Пример 1. Задачу Коши для дифференциального уравнения второго порядка

$$y'' + y = 0, \quad y|_{x=0}=0, \quad y'|_{x=0}=1$$

преобразовать к задаче Коши для системы двух дифференциальных уравнений первого порядка.

Решение. Положим $y_1 = y$, $y_2 = y'_1$; тогда функцию $y(x)$ и ее производные до второго порядка включительно можно выразить через y_1 , y_2 , y'_1 , y'_2 :

$$y = y_1, \quad y' = y'_1 = y_2, \quad y'' = y'_2.$$

Подставляя новые переменные, приходим к задаче Коши для системы уравнений, решенной в примере из §2, поскольку

$$\begin{cases} y'_1 = y_2 - \text{уравнение, полученное при введении функций } y_1, y_2; \\ y'_2 = -y_1 - \text{преобразованное исходное уравнение } y'' = -y; \end{cases}$$

$$y_1|_{x=0} = 0, \quad y_2|_{x=0} = 1. \quad \blacksquare$$

На практике возникают задачи Коши для систем дифференциальных уравнений, порядок которых выше первого. Например, при решении задач динамики механических систем может получиться система дифференциальных уравнений, состоящая только из уравнений второго порядка. Покажем на примере, как задача Коши для системы дифференциальных уравнений высших порядков преобразуется к задаче Коши для системы уравнений первого порядка.

Пример 2. Задачу Коши для системы двух дифференциальных уравнений второго порядка

$$\begin{cases} y''_1 + y'_1 + 3(y_1 - y_2) = \sin x, \\ y''_2 + y'_2 + 5(y_2 - y_1) = 0, \end{cases}$$

$$y_1|_{x=0}=1, \quad y'_1|_{x=0}=0, \quad y_2|_{x=0}=0, \quad y'_2|_{x=0}=1$$

преобразовать к задаче Коши для системы четырех дифференциальных уравнений первого порядка. Найти решение этой задачи методом Рунге – Кутты на сетке отрезка $[0; 0,4]$ с шагом 0,2 и оценить погрешность полученного результата с помощью правила Рунге.

Решение. Введем функции z_1, z_2, z_3, z_4 :

$$z_1 = y_1, \quad z_2 = y_1', \quad z_3 = y_2, \quad z_4 = y_2'.$$

Тогда получим задачу Коши для системы дифференциальных уравнений первого порядка:

$$\begin{cases} z_1' = z_2, \\ z_2' = -z_2 - 3(z_1 - z_3) + \sin x, \\ z_3' = z_4, \\ z_4' = -z_4 - 5(z_3 - z_1), \end{cases}$$

$$z_1|_{x=0}=1, \quad z_2|_{x=0}=0, \quad z_3|_{x=0}=0, \quad z_4|_{x=0}=1.$$

Численное решение этой задачи, найденное с помощью программы на языке QUICKBASIC в Приложении (см. с. 325), представлено в таблице. Для оценки погрешности по правилу Рунге решение определялось дважды методом Рунге - Кутты: один раз с шагом 0,2, другой - с шагом 0,1.

t	x_t	Численное решение задачи Коши			
		с шагом 0,2		с шагом 0,1	
		$y_{1t}=z_{1t}$	$y_{2t}=z_{3t}$	$y_{1t}=z_{1t}$	$y_{2t}=z_{3t}$
0	0	1	0	1	0
1	0,2	0,950265	0,266267	0,950106	0,266537
2	0,4	0,844404	0,604998	0,844097	0,605519

Погрешность определяется величиной

$$\frac{\max_{1 \leq t \leq 2} \left(\max_{1 \leq k \leq 2} \left(|y_{kt}(h) - y_{kt}(h/2)| \right) \right)}{15} =$$

$$= \frac{|y_{22}(h) - y_{22}(h/2)|}{15} = \frac{0,000521}{15} = 0,000035. \quad \blacksquare$$

Упражнения

Задачу Коши для данного дифференциального уравнения второго порядка преобразовать к задаче Коши для системы дифференциальных уравнений первого порядка. Найти решение последней задачи методом Рунге - Кутты на сетке отрезка $[a, b]$. Вычисления про-

вести дважды с шагами h и $h/2$, полагая $h=0,2$. Найти численное решение дифференциального уравнения и оценить его погрешность с помощью правила Рунге. Сравнить численное решение с известным аналитическим решением $\varphi(x)$. Результаты представить в виде таблицы, аналогичной таблице на с.144. Относительно иррациональных исходных данных задачи учесть замечания к упражнениям из §1.

1. $xy'' - (x+1)y' - 2(x-1)y = 0$, $y|_{x=1} = e^2$, $y'|_{x=1} = 2e^2$, $1 \leq x \leq 2$, $\varphi = e^{2x}$.
2. $x^2y'' + xy' - y = 3x^2$, $y|_{x=1} = 1$, $y'|_{x=1} = 1$, $1 \leq x \leq 2$, $\varphi = \frac{1}{2}(2x^2 - x + 1/x)$.
3. $x^2y'' - 6y = 0$, $y|_{x=1} = 1$, $y'|_{x=1} = 3$, $1 \leq x \leq 2$, $\varphi(x) = x^3$.
4. $x^2y'' - 12y = 0$, $y|_{x=1} = 1$, $y'|_{x=1} = 4$, $1 \leq x \leq 2$, $\varphi(x) = x^4$.
5. $xy'' + \frac{1}{2}y' = 0$, $y|_{x=1} = 2$, $y'|_{x=1} = 1$, $1 \leq x \leq 2$, $\varphi(x) = 2\sqrt{x}$.
6. $x^2y'' + y/\ln x = xe^x(2+x\ln x)$, $y|_{x=2} = e^2\ln 2$, $y'|_{x=2} = e^2(\ln 2 + 0,5)$,
 $2 \leq x \leq 3$, $\varphi(x) = e^x \ln x$.
7. $x^2y'' + xy' = 0$, $y|_{x=1} = 0$, $y'|_{x=1} = 1$, $1 \leq x \leq 2$, $\varphi(x) = \ln x$.
8. $x^2y'' - xy' + y = 3x^3$, $y|_{x=1} = 0,75$, $y'|_{x=1} = 2,25$, $1 \leq x \leq 2$, $\varphi(x) = \frac{3}{4}x^3$.
9. $x^2y'' - 4xy' + 6y = x^4 - x^2$, $y|_{x=1} = 0,5$, $y'|_{x=1} = 3$, $1 \leq x \leq 2$,
 $\varphi(x) = \frac{x^4}{2} + x^2 \ln x$.
10. $x^2y'' + (x^2 - 1)y' - y = 0$, $y|_{x=1} = \frac{1}{e}$, $y'|_{x=1} = -\frac{1}{e}$, $1 \leq x \leq 2$, $\varphi(x) = e^{-x}$.
11. $x^2y'' - (x^2 - 2x)y' - (3x + 2)y = 0$, $y|_{x=1} = e$, $y'|_{x=1} = 2e$, $1 \leq x \leq 2$,
 $\varphi(x) = xe^x$.
12. $x^2y'' + x^3y' + (x^2 - 2)y = 0$, $y|_{x=1} = 1$, $y'|_{x=1} = -1$, $1 \leq x \leq 2$, $\varphi(x) = \frac{1}{x}$.

$$13. (x^2-1)y''-2xy'+2y=0, \quad y|_{x=2}=5, \quad y'|_{x=2}=4, \quad 2 \leq x \leq 3, \quad \varphi(x)=x^2+1.$$

$$14. x(x+1)y''+(3x+2)y'+y=0, \quad y|_{x=1}=1, \quad y'|_{x=1}=-1, \quad 1 \leq x \leq 2, \quad \varphi(x)=\frac{1}{x}.$$

$$15. y''-2(2x^2+1)y=0, \quad y|_{x=0}=1, \quad y'|_{x=0}=0, \quad 0 \leq x \leq 1, \quad \varphi(x)=e^{x^2}.$$

$$16. y''+y'\operatorname{tg}x-y\cos^2x=0, \quad y|_{x=0}=1, \quad y'|_{x=0}=-1, \quad 0 \leq x \leq 1, \quad \varphi(x)=e^{-\sin x}.$$

$$17. y''+4xy'+(4x^2+2)y=0, \quad y|_{x=0}=0, \quad y'|_{x=0}=1, \quad 0 \leq x \leq 1, \quad \varphi(x)=xe^{x^2}.$$

$$18. y''+2y'\operatorname{tg}x+3y=0, \quad y|_{x=0}=1, \quad y'|_{x=0}=0, \quad 0 \leq x \leq 1, \quad \varphi(x)=\cos^3x.$$

$$19. x(x+1)y''-(x-1)y'+y=0, \quad y|_{x=1}=y'|_{x=1}=-4, \quad 1 \leq x \leq 2, \\ \varphi(x)=(x-1)\ln|x|-4x.$$

$$20. y''+4y=\cos^3x, \quad y|_{x=0}=\frac{1}{5}, \quad y'|_{x=0}=0, \quad 0 \leq x \leq 1, \\ \varphi(x)=\frac{1}{4}\left[\cos x-\frac{\cos 3x}{5}\right].$$

$$21. y'' \cdot x^2 \ln x - xy' + y = 0, \quad y|_{x=1}=1, \quad y'|_{x=1}=1, \quad 1 \leq x \leq 2, \quad \varphi(x)=\ln x + 1.$$

$$22. (e^x+1)y''-2y'-e^xy=0, \quad y|_{x=0}=0, \quad y'|_{x=0}=1, \quad 0 \leq x \leq 1, \quad \varphi(x)=e^x-1.$$

$$23. y'' \sin y - 2(y')^2 \cos y = 0, \quad y|_{x=0}=\frac{\pi}{4}, \quad y'|_{x=0}=\frac{1}{2}, \quad 0 \leq x \leq 1, \\ \varphi(x)=\operatorname{arccotg}(1-x).$$

$$24. (x-2)^2y''-3(x-2)y'+4y=x+2, \quad y|_{x=3}=3, \quad y'|_{x=3}=3, \quad 3 \leq x \leq 4, \\ \varphi(x)=(x-2)^2+x-1.$$

$$25. y''-2y=4x^2e^{x^2}, \quad y|_{x=0}=1, \quad y'|_{x=0}=0, \quad 0 \leq x \leq 1, \quad \varphi(x)=e^{x^2}.$$

ЧИСЛЕННЫЕ МЕТОДЫ БЕЗУСЛОВНОЙ ОПТИМИЗАЦИИ

**§1. НЕОБХОДИМЫЕ И ДОСТАТОЧНЫЕ УСЛОВИЯ ЭКСТРЕМУМА
В ЗАДАЧАХ БЕЗУСЛОВНОЙ ОПТИМИЗАЦИИ**

Пусть задано множество $X \subset R^n$ и функция $f(x) = f(x_1, x_2, \dots, x_n)$, определенная на множестве X .

Определения. Точка $x^* \in X$ называется **точкой локального минимума** функции $f(x)$ на множестве X , если существует шар $U_\varepsilon(x^*) = \{x: \|x - x^*\| \leq \varepsilon\}$ такой, что для любого $x \in U_\varepsilon(x^*)$ выполняется неравенство

$$f(x^*) \leq f(x). \quad (1)$$

Если неравенство (1) выполняется как строгое (при $x \neq x^*$), то говорят, что x^* — **точка строгого локального минимума**.

Точка $x^* \in X$ называется **точкой глобального минимума** функции $f(x)$ на множестве X , если неравенство (1) выполняется для любого x из множества X .

Аналогично определяются точки локального и глобального максимума функции $f(x)$ на множестве X .

Точки локального минимума и максимума функции $f(x)$ называют **точками экстремума** этой функции.

Задача отыскания всех локальных минимумов (максимумов) функции $f(x)$, если множество X совпадает со всем n -мерным пространством, т.е. $X = R^n$, называется **задачей безусловной оптимизации**, а функция $f(x)$ — **целевой функцией**.

Задачу отыскания точек локального минимума целевой функции $f(x)$ символически записывают так:

$$f(x) \rightarrow \min, x \in R^n. \quad (2)$$

Аналогично, задачу отыскания точек локального максимума функции $f(x)$ символически записывают следующим образом:

$$f(x) \rightarrow \max, x \in R^n. \quad (3)$$

Задача (3) эквивалентна задаче

$$-f(x) \rightarrow \min, x \in R^n$$

в том смысле, что множества локальных и глобальных решений этих задач соответственно совпадают.

Теорема 1 (о необходимых условиях локального минимума). Пусть x^* — точка локального минимума функции $f(x)$, которая имеет

в этой точке непрерывные частные производные $\frac{\partial f}{\partial x_j}(\mathbf{x})$ ($j=1,2,\dots,n$); тогда частные производные функции $f(\mathbf{x})$ в этой точке равны нулю, т.е.

$$\frac{\partial f}{\partial x_j}(\mathbf{x}^*) = 0 \quad (j=1,2,\dots,n).$$

Иначе говоря, в этой точке градиент функции

$$\nabla f(\mathbf{x}^*) = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right]$$

равен нулевому вектору, т.е. $\nabla f(\mathbf{x}^*) = 0$.

Определение. Точке $\mathbf{x}^*$, удовлетворяющая условию

$$\nabla f(\mathbf{x}^*) = 0,$$

называется **стационарной точкой** функции $f(\mathbf{x})$.

Не всякая стационарная точка функции $f(\mathbf{x})$ является решением задачи (2).

Определения. Пусть даны квадратная матрица $A=(a_{ij})$ порядка n и вектор $\mathbf{x} \in \mathbb{R}^n$. Умножение матрицы A на вектор $\mathbf{x}$ дает вектор $\mathbf{y}=\mathbf{Ax}$, координаты которого определяются соотношениями

$$y_i = \sum_{j=1}^n a_{ij}x_j \quad (i=1,2,\dots,n).$$

Квадратная матрица A называется **симметричной**, если $a_{ij}=a_{ji}$.

Симметричная матрица A называется **неотрицательно определенной**, если для любого вектора $\mathbf{x} \in \mathbb{R}^n$ скалярное произведение векторов $\mathbf{y}=\mathbf{Ax}$ и $\mathbf{x}$ неотрицательно, т.е. $(\mathbf{Ax}, \mathbf{x}) \geq 0$; **положительно определенной**, если $(\mathbf{Ax}, \mathbf{x}) > 0$, $\mathbf{x} \neq 0$; **неположительно определенной**, если $(\mathbf{Ax}, \mathbf{x}) \leq 0$; **отрицательно определенной**, если $(\mathbf{Ax}, \mathbf{x}) < 0$, $\mathbf{x} \neq 0$.

Теорема 2 (критерий Сильвестра). Симметричная матрица A неотрицательно (положительно) определена тогда и только тогда, когда все главные (угловые) миноры неотрицательны (положительны):

$$\Delta_1 = a_{11} \geq 0 \quad (> 0), \quad \Delta_2 = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \geq 0 \quad (> 0),$$

$$\Delta_3 = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} \geq 0 \quad (> 0), \dots, \det A \geq 0 \quad (> 0).$$

Симметричная матрица A является неположительно (отрицательно) определенной тогда и только тогда, когда знаки последовательных главных миноров чередуются, причем $\Delta_1 = a_{11} \leq 0 \quad (< 0)$,

$$\Delta_2 = \geq 0 \quad (> 0), \quad \Delta_3 = \leq 0 \quad (< 0), \dots$$

Определение. Матрица вида

$$H(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

называется матрицей Гессе функции $f(x)$.

Теорема 3. Если точка x^* — локальное решение задачи (2) и в этой точке функция $f(x)$ имеет непрерывные частные производные до второго порядка включительно, то матрица Гессе функции $f(x)$ в точке x^* является неотрицательно определенной, т.е.

$$(H(x^*)x, x) \geq 0 \quad \forall x \in R^n.$$

Теорема 4 (о достаточных условиях локального экстремума). Если точка x^* является стационарной точкой функции $f(x)$, т.е. $\nabla f(x^*) = 0$, и матрица Гессе функции $f(x)$ в точке x^* положительно определена, то x^* — строгое локальное решение задачи (2); если же точка x^* является стационарной для функции $f(x)$ и матрица Гессе в ней отрицательно определена, то x^* — строгое локальное решение задачи (3).

Замечание. Задача одномерной минимизации

$$f(x) \rightarrow \min, \quad x \in R$$

является частным случаем задачи (2).

Для этого частного случая теорема 4 формулируется следующим образом.

Теорема 4'. Пусть функция $f(x)$ имеет в точке $x^* \in \mathbb{R}$ непрерывные производные до второго порядка включительно. Тогда если x^* — стационарная точка функции $f(x)$, т.е. $f'(x^*) = 0$ и $\frac{d^2 f}{dx^2}(x^*) > 0$, то x^* — строгое локальное решение задачи одномерной минимизации.

Пример 1. Решить задачу

$$f(x) = x_1^3 + x_2^3 - 3x_1x_2 \rightarrow \min, \quad x \in \mathbb{R}^2.$$

Решение. Найдем стационарные точки функции $f(x)$:

$$\frac{\partial f}{\partial x_1} = 3x_1^2 - 3x_2 = 0, \quad \frac{\partial f}{\partial x_2} = 3x_2^2 - 3x_1 = 0.$$

Решением этой системы уравнений являются две точки

$$x_1^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad x_2^* = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

При этом $H(x) = \begin{bmatrix} 6x_1 & -3 \\ -3 & 6x_2 \end{bmatrix}$, $H(x_1^*) = \begin{bmatrix} 0 & -3 \\ -3 & 0 \end{bmatrix}$, $H(x_2^*) = \begin{bmatrix} 6 & -3 \\ -3 & 6 \end{bmatrix}$.

В силу критерия Сильвестра матрица $H(x_1^*)$ не является неотрицательно определенной, а матрица $H(x_2^*)$ положительно определена. Следовательно, на основании теоремы 3 точка x_1^* не является решением задачи; согласно теореме 4, в точке x_2^* данная целевая функция достигает строгого локального минимума. ■

Пример 2. Исследовать на экстремум функцию

$$f(x) = \frac{1}{2} x_1 x_2 + (47 - x_1 - x_2) \left(\frac{x_1}{3} + \frac{x_2}{4} \right).$$

Решение. Найдем стационарные точки:

$$\frac{\partial f}{\partial x_1} = -\frac{1}{12} x_2 - \frac{2}{3} x_1 + \frac{47}{3} = 0, \quad \frac{\partial f}{\partial x_2} = -\frac{1}{2} x_2 - \frac{1}{12} x_1 + \frac{47}{4} = 0$$

или
$$\begin{cases} 8x_1 + x_2 = 188, \\ x_1 + 6x_2 = 141. \end{cases}$$

Решением этой системы является точка $\mathbf{x}^* = \begin{bmatrix} 21 \\ 20 \end{bmatrix}$.

При этом

$$H(\mathbf{x}^*) = \begin{bmatrix} -\frac{2}{3} & -\frac{1}{12} \\ \frac{1}{12} & -\frac{1}{2} \end{bmatrix}, \quad \Delta_1 = a_{11} = -\frac{2}{3} < 0, \quad \Delta_2 = \frac{1}{3} - \frac{1}{144} > 0.$$

Следовательно, в точке $\mathbf{x}^*$ имеет место строгий локальный максимум. ■

§2. ВЫПУКЛЫЕ МНОЖЕСТВА И ВЫПУКЛЫЕ ФУНКЦИИ

Определение. Множество $X \subset \mathbb{R}^n$ называется **выпуклым**, если вместе с любыми двумя точками $\mathbf{x}_1$ и $\mathbf{x}_2$ ему целиком принадлежит отрезок, соединяющий эти точки (рис.26).

Аналитически условие выпуклости множества записывается так:

$$\mathbf{x} = (1-\alpha)\mathbf{x}_1 + \alpha\mathbf{x}_2, \quad \mathbf{x} \in X, \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in X, \quad \alpha \in [0, 1]. \quad (4)$$

Определение. Функция $f(\mathbf{x})$, определенная на выпуклом множестве $X \subset \mathbb{R}^n$, называется **выпуклой**, если $\forall \mathbf{x}_1, \mathbf{x}_2 \in X, \alpha \in [0, 1]$ справедливо неравенство

$$f((1-\alpha)\mathbf{x}_1 + \alpha\mathbf{x}_2) \leq (1-\alpha)f(\mathbf{x}_1) + \alpha f(\mathbf{x}_2). \quad (5)$$

Если неравенство (5) выполняется как строгое, то функция $f(\mathbf{x})$ называется **строго выпуклой** на множестве X .

Определение выпуклой функции одной переменной иллюстрирует рис.27. Через любые две точки $(x_1, f(x_1))$ и $(x_2, f(x_2))$, лежащие на кривой $y=f(x)$, проведем хорду. Параметрические уравнения этой хорды имеют вид

$$\begin{cases} x = (1-\alpha)x_1 + \alpha x_2, \\ y = (1-\alpha)f(x_1) + \alpha f(x_2), \end{cases} \quad 0 \leq \alpha \leq 1.$$

Формальное определение выпуклой функции одной переменной в виде неравенства (5), в частности, означает, что для аргумента $x \in [x_1, x_2]$ ордината графика данной функции $f(x) = f((1-\alpha)x_1 + \alpha x_2)$ не больше ординаты хорды $(1-\alpha)f(x_1) + \alpha f(x_2)$. График функции является выпуклым вниз по отношению к любой его хорде.

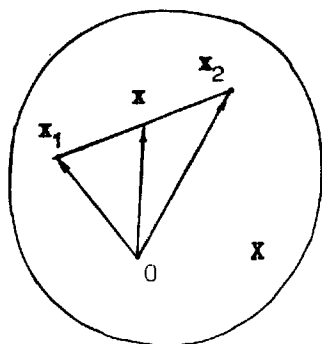


Рис.26

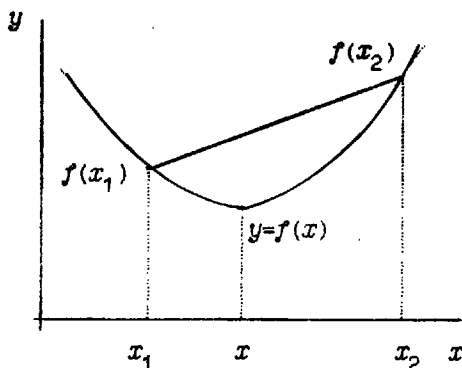


Рис.27

Теорема 5. Если функция $f(\mathbf{x})$ выпукла на множестве X и $\mathbf{x}^*$ является стационарной точкой функции $f(\mathbf{x})$, т.е. $\nabla f(\mathbf{x}^*) = 0$, то $\mathbf{x}^*$ — строгое локальное решение задачи

$$f(\mathbf{x}) \rightarrow \min, \mathbf{x} \in X. \quad (6)$$

Теорема 6. Если функция $f(\mathbf{x})$ и множество X выпуклы, то любое локальное решение задачи (6) является также глобальным решением на множестве X .

Пусть функция $f(\mathbf{x})$ выпукла на множестве X ; тогда из теорем 5 и 6 следует, что отыскание стационарной точки задачи (6) означает нахождение глобального решения этой задачи.

Достаточные условия выпуклости функции содержит следующая теорема.

Теорема 7. Если функция $f(\mathbf{x})$ имеет непрерывные производные до второго порядка включительно и матрица Гессе функции $f(\mathbf{x})$ положительно определена в любой точке $\mathbf{x}$ выпуклого множества X , то функция $f(\mathbf{x})$ является выпуклой на множестве X .

Пример 1. Показать, что стационарная точка функции

$$f(\mathbf{x}) = x_1^2 - x_1 x_2 + x_2^2 + 9x_1 - 6x_2 + 20$$

является глобальным решением задачи $f(\mathbf{x}) \rightarrow \min, \mathbf{x} \in \mathbb{R}^2$.

Решение. Найдем стационарную точку функции $f(\mathbf{x})$:

$$\frac{\partial f}{\partial x_1} = 2x_1 - x_2 + 9 = 0, \quad \frac{\partial f}{\partial x_2} = -x_1 + 2x_2 - 6 = 0.$$

Решением этой системы уравнений является точка $\mathbf{x}^* = \begin{bmatrix} -4 \\ 1 \end{bmatrix}$.

Вычислим частные производные второго порядка и составим матрицу Гессе данной функции:

$$\frac{\partial^2 f}{\partial x_1^2} = 2, \quad \frac{\partial^2 f}{\partial x_2^2} = 2, \quad \frac{\partial^2 f}{\partial x_1 \partial x_2} = \frac{\partial^2 f}{\partial x_2 \partial x_1} = -1; \quad \mathcal{H}(\mathbf{x}) = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}.$$

Угловые миноры $\Delta_1=2$ и $\Delta_2=3$ матрицы $\mathcal{H}(\mathbf{x})$ не зависят от координат точки и положительны во всех точках выпуклого множества $\mathbb{R}^2$. Согласно критерию Сильвестра, матрица Гессе положительно определена на множестве $\mathbb{R}^2$, и, следовательно, функция является выпуклой на $\mathbb{R}^2$, а единственная стационарная точка $\mathbf{x}^*$ — решение глобальной задачи. ■

Пример 2. Найти локальный минимум функции

$$f(\mathbf{x}) = 3x_1^2 - 2x_1\sqrt{x_2} + x_2 - 8x_1 + 8.$$

На каком множестве найденное локальное решение является глобальным?

Решение. Найдем стационарную точку:

$$\frac{\partial f}{\partial x_1} = 6x_1 - 2\sqrt{x_2} - 8 = 0, \quad \frac{\partial f}{\partial x_2} = -\frac{x_1}{\sqrt{x_2}} + 1 = 0.$$

Решение системы — точка $\mathbf{x}^* = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$. Составим матрицу Гессе:

$$\frac{\partial^2 f}{\partial x_1^2} = 6, \quad \frac{\partial^2 f}{\partial x_2^2} = \frac{x_1}{2x_2\sqrt{x_2}}, \quad \frac{\partial^2 f}{\partial x_1 \partial x_2} = \frac{\partial^2 f}{\partial x_2 \partial x_1} = -\frac{1}{\sqrt{x_2}},$$

$$\mathcal{H}(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 6 & -\frac{1}{\sqrt{x_2}} \\ -\frac{1}{\sqrt{x_2}} & \frac{x_1}{2x_2\sqrt{x_2}} \end{bmatrix}.$$

Матрица Гессе положительно определена на множестве точек $\mathbf{x} \in X$, где угловые миноры $\mathcal{H}(\mathbf{x})$ положительны, т.е.

$$\Delta_1 = 6 > 0 \quad \text{и} \quad \Delta_2 = \frac{3x_1 - \sqrt{x_2}}{x_2 \sqrt{x_2}} > 0.$$

Отсюда следует, что $X = \{(x_1, x_2): x_2 > 0, x_1 > \frac{\sqrt{x_2}}{3}\}$. Непосредственной проверкой убеждаемся, что стационарная точка x^* принадлежит множеству X и, следовательно, согласно теореме 6, x^* является точкой глобального минимума функции $f(x)$ на множестве X . ■

Упражнения

Найти решение задачи безусловной минимизации

$$f(x) \rightarrow \min, x \in R^2$$

для заданной целевой функции $f(x)$, используя теоремы о необходимых и достаточных условиях безусловной минимизации. Установить, на каком множестве найденное решение является глобальным.

1. $f(x) = x_1^3 + 8x_2^3 - 6x_1x_2 + 1.$
2. $f(x) = e^{x_1/2} (x_1 + x_2^2).$
3. $f(x) = x_1^2 + x_2^2 + x_1x_2 - 3x_1 - 6x_2.$
4. $f(x) = x_1^2 + x_2^2 + 4(x_2 - x_1).$
5. $f(x) = x_1^2 + x_2^2 + x_1x_2 + x_1 - x_2 + 1.$
6. $f(x) = x_1^3 + x_2^2 - 6x_1x_2 - 39x_1 + 18x_2 + 20.$
7. $f(x) = (x_1 - 3)^2 + (x_2 - 2)^2 + (x_1 - x_2 - 4)^2.$
8. $f(x) = 2x_1^2 + x_2^3 - 4x_1 - 3x_2 + 6.$
9. $f(x) = x_1^2 + x_2^2 + x_1x_2 - 3x_1 - 6x_2.$
10. $f(x) = x_1^2 + x_2^2 - 4\sqrt{x_1x_2} - 2x_1 - 2x_2 + 8.$
11. $f(x) = 2x_1^3 - x_1x_2^2 + 5x_1^2 + x_2^2.$
12. $f(x) = 2x_1^2 + x_2^2 + 4(x_2 - x_1) + 6.$

$$13. f(x) = x_1^2 + x_2^2 + x_1 x_2 - 3x_1 - 6x_2.$$

$$14. f(x) = 8x_1^2 + 4x_1 x_2 + 2x_2^2 + 4x_1 - 4x_2.$$

$$15. f(x) = x_1 x_2 + \frac{50}{x_1} + \frac{20}{x_2}.$$

$$16. f(x) = x_1^2 + 2x_2^2 + 2(x_1 + 4x_2) + 5.$$

$$17. f(x) = x_1^2 + x_2^2 - 2 \cdot \ln x_1 - 18 \cdot \ln x_2.$$

$$18. f(x) = x_1^3 + x_2^3 - 15x_1 x_2.$$

$$19. f(x) = (x_1^2 + x_2^2)^{2/3} - 4.$$

$$20. f(x) = (x_1^2 + x_2^2) \left[e^{-(x_1^2 + x_2^2)} - 1 \right].$$

$$21. f(x) = x_1^4 - 2x_1^2 + 5x_2^2 - 2x_1^2 x_2 - 2x_2 + 2 \quad (x_1 > 0).$$

$$22. f(x) = x_1^4 - 2x_1^2 + 5x_2^2 - 2x_1^2 x_2 - 2x_2 + 2 \quad (x_1 < 0).$$

$$23. f(x) = x_1^4 + 2x_2^2 - 2x_1^2 x_2 - 2x_2 + 1 \quad (x_1 > 0).$$

$$24. f(x) = x_1^4 + 2x_2^2 - 2x_1^2 x_2 - 2x_2 + 1 \quad (x_1 < 0).$$

$$25. f(x) = x_1^2 + 3x_2^2 + 4x_1 - 6x_2 + 7.$$

§3. ЧИСЛЕННЫЕ МЕТОДЫ ПОИСКА МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ

1⁰. УНИМОДАЛЬНЫЕ ФУНКЦИИ

Определение. Непрерывная функция $y=f(x)$ называется **униmodalной** на отрезке $[a, b]$, если:

- 1) точка x^* локального минимума функции принадлежит отрезку $[a, b]$;
- 2) для любых двух точек отрезка x_1 и x_2 , взятых по одну сторону от точки минимума, точке x_1 , более близкой к точке минимума, соответствует меньшее значение функции, т.е. как при $x^* < x_1 < x_2$ (рис.28,а), так и при $x_2 < x_1 < x^*$ (рис.28,б) справедливо неравенство $f(x_1) < f(x_2)$.

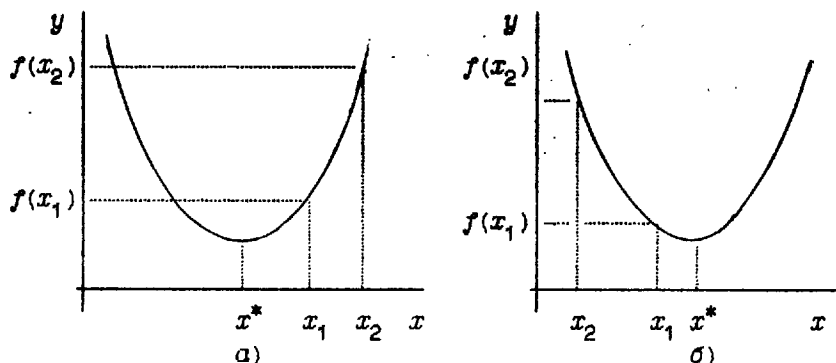


Рис.28

Достаточное условие унимодальности функции $f(x)$ на отрезке $[a, b]$ содержится в следующей теореме.

Теорема 8. Если функция $f(x)$ дважды дифференцируема на отрезке $[a, b]$ и $f''(x) > 0$ в любой точке этого отрезка, то $f(x)$ — унимодальная функция на $[a, b]$.

Заметим, что условие $f''(x) > 0$ определяет множество точек, на котором функция является выпуклой вниз.

Пример 1. Для функции $f(x) = 2x^2 - \ln x$ найти:

- 1) промежуток X , на котором функция является унимодальной;
- 2) решение задачи $f(x) \rightarrow \min, x \in X (X \subset \mathbb{R})$.

Решение. Функция $f(x)$ определена при $x > 0$; найдем ее производные

$$f'(x) = \frac{4x^2 - 1}{x}, \quad f''(x) = \frac{4x^2 + 1}{x^2} > 0, \quad x \in (0, \infty).$$

Следовательно, функция $f(x)$ унимодальна на интервале $(0, \infty)$. Первая производная $f'(x) = 0$ при $x = 0,5$. На основании теоремы 4' заключаем, что $x^* = 0,5$ — точка локального минимума функции $f(x)$. ■

2°. Схема сужения промежутка унимодальности функции

Пусть требуется решить задачу

$$f(x) \rightarrow \min, \quad x \in X \quad (X \subset \mathbb{R}). \quad (7)$$

Применение численных методов для отыскания точек x^* локального минимума функции $f(x)$ предполагает:

- 1) определение промежутков унимодальности функции, т.е. на-

хождение отрезков, которым принадлежит одна точка локального минимума;

2) вычисление значения x^* , принадлежащего выбранному промежутку, с заданной точностью.

Для непрерывной функции $f(x)$ строят ее график на некотором отрезке $[a, b]$ и если окажется, что на этом отрезке график функции имеет вид, изображенный на рис. 28, то $[a, b]$ — отрезок унимодальности функции. Отрезок $[a, b]$ берется по возможности малым.

При вычислении точки минимума точность достигается последовательным уменьшением отрезка $[a, b]$, содержащего точку x^* , до размеров, не превышающих заданную точность ε ($b - a \leq \varepsilon$).

Замечание. Если функция имеет производную в области определения, то для отыскания стационарных точек функции $f(x)$ нужно найти корни уравнения $f'(x) = 0$. Для решения этого уравнения, как правило, необходимо использовать численные методы, описанные в гл.3. Однако для решения задачи (7) проще применять прямые численные методы поиска минимума функции $f(x)$.

Рассмотрим один из приемов, позволяющих сузить отрезок унимодальности функции.

Пусть функция $f(x)$ унимодальна на отрезке $[a, b]$. Возьмем две произвольные точки x_1 и x_2 , принадлежащие отрезку $[a, b]$ и такие, что $x_1 < x_2$. В каждом из следующих трех очевидных возможных случаев можно указать отрезок меньших размеров $[a_1, b_1]$, содержащий точку минимума x^* и принадлежащий первоначальному отрезку (рис.29):

I. Если $y_1 = f(x_1) < y_2 = f(x_2)$, то положим $a_1 = a$ и $b_1 = x_2$ и получим меньший отрезок унимодальности $[a_1, b_1]$.

II. Если $y_1 = f(x_1) > y_2 = f(x_2)$, то естественно принять $a_1 = x_1$ и $b_1 = b$.

III. Если $y_1 = f(x_1) = y_2 = f(x_2)$, то $a_1 = x_1$ и $b_1 = x_2$.

Пример 2. Для функции $f(x) = 2x^2 - \ln x$, выбрав отрезок унимодальности $[a, b] = [0,25; 1]$ и две произвольные точки

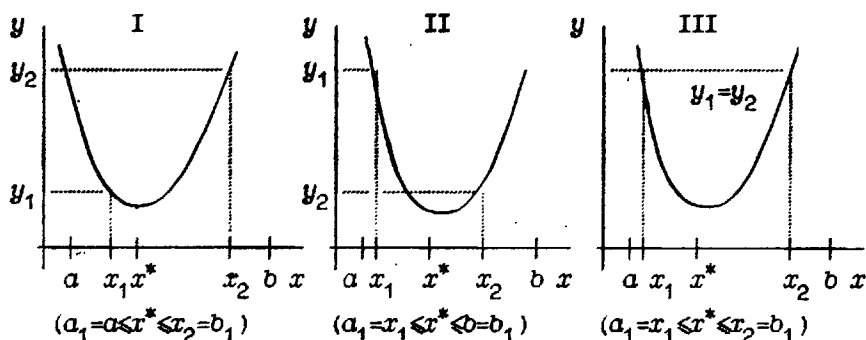


Рис. 29

$x_1, x_2 \in [a, b]$, найти меньший отрезок унимодальности $[a_1, b_1]$.

Решение. В примере 1 было установлено, что функция $f(x)$ имеет точку минимума $x^* = 0,5$ и является унимодальной на любом отрезке, содержащем точку x^* и лежащем в области ее определения $(0, \infty)$. Возьмем $x_1 = 0,375$, $x_2 = 0,625$; тогда $y_1 = f(x_1) = 1,2620793$, $y_2 = f(x_2) = 1,2512536$. Здесь естественно положить $a_1 = x_1 = 0,375$ и $b_1 = b = 1$ (случай II). ■

Методы вычисления значения точки минимума функции одной переменной отличаются алгоритмами выбора точек x_1 и x_2 для локализации точки x^* с заданной точностью.

3°. Метод половинного деления

Пусть при решении задачи (7) определен отрезок $[a, b]$, которому принадлежит точка локального минимума x^* , и функция $f(x)$ является унимодальной на этом отрезке.

Далее для сужения отрезка унимодальности используем точки x_1 и x_2 , расположенные симметрично относительно середины отрезка $[a, b]$:

$$x_{1,2} = \frac{a+b}{2} \mp k \frac{b-a}{2}.$$

Будем считать, что число k гораздо меньше 1 ($k \ll 1$). Тогда точки x_1 и x_2 принадлежат отрезку $[a, b]$ и, следуя рассмотренной в п. 2° схеме, получим новый суженный отрезок $[a_1, b_1]$ и оценим

его длину в каждом из трех возможных случаев:

$$\text{I. } y_1 < y_2, \quad a_1 = a, \quad b_1 = x_2 = \frac{a+b}{2} + k \frac{b-a}{2}; \quad b_1 - a_1 = \frac{1+k}{2}(b-a).$$

$$\text{II. } y_1 > y_2, \quad a_1 = x_1 = \frac{a+b}{2} - k \frac{b-a}{2}, \quad b_1 = b; \quad b_1 - a_1 = \frac{1+k}{2}(b-a).$$

$$\text{III. } y_1 = y_2, \quad a_1 = x_1, \quad b_1 = x_2; \quad b_1 - a_1 = k(b-a).$$

Таким образом, после первого шага преобразований найден новый отрезок унимодальности $[a_1, b_1]$, длина которого уменьшилась.

Название метода (метод половинного деления) мотивировано тем, что если величина k очень мала, то длина отрезка унимодальности $b-a$ уменьшается почти вдвое (в случаях I, II).

Теперь в новом суженном промежутке $[a_1, b_1]$ выберем точки $x_1^{(1)}$ и $x_2^{(1)}$, симметричные относительно его середины:

$$x_{1,2}^{(1)} = \frac{a_1 + b_1}{2} \mp k \frac{b_1 - a_1}{2}.$$

Проведя вычисления, аналогичные сделанным ранее, получаем отрезок $[a_2, b_2]$, длина которого не больше

$$b_2 - a_2 = (1+k) \frac{b_1 - a_1}{2} = (1+k)^2 \frac{b-a}{2^2},$$

и т. д.

В результате приходим к последовательности таких вложенных отрезков $[a, b]$, $[a_1, b_1]$, ..., $[a_n, b_n]$, ..., что точка локального минимума x^* функции $f(x)$ принадлежит каждому из них и является общим пределом последовательностей $\{a_n\}$ и $\{b_n\}$.

Отсюда получаются приближенные равенства

$$x^* \approx a_n \approx b_n,$$

точность которых на n -м шаге вычислений можно оценить неравенством

$$0 \leq x^* - a_n \leq b_n - a_n \leq \frac{(1+k)^n}{2^n} (b-a) < \varepsilon. \quad (8)$$

Пример 3. Найти точку x^* локального минимума функции

$f(x) = 2x^2 - \ln x$ на отрезке $[0,25; 1]$ методом половинного деления с точностью $\varepsilon = 0,1$. Провести вычисления, полагая $k = 0,1$ и предварительно оценив минимальное число шагов n для достижения точности ε .

Решение. Было установлено (см. пример 1), что функция унимодальна на отрезке $[0,25; 1]$; точка x^* принадлежит этому отрезку. Воспользуемся неравенством (8) и определим число шагов n :

$$\frac{(1+k)^n}{2^n} (b-a) < \varepsilon; \quad n > \frac{\ln[\varepsilon/(b-a)]}{\ln[(1+k)/2]} = \frac{\ln(0,1/0,75)}{\ln(0,55)} \approx 3,37; \quad n_{\min} = 4.$$

Обозначим

$$x_{1,2}^{(t)} = \frac{a_t + b_t}{2} \mp k \frac{b_t - a_t}{2}, \quad y_1^{(t)} = f(x_1^{(t)}), \quad y_2^{(t)} = f(x_2^{(t)}) \quad (9)$$

$$(t = 0, 1, 2, \dots, n),$$

где $a_0 = a = 0,25$, $b_0 = b = 1$, a_t, b_t — координаты начала и конца отрезка, полученные на t -м шаге вычислений, точки $x_1^{(t)}$ и $x_2^{(t)}$ принадлежат отрезку $[a_t, b_t]$.

Произведем последовательные вычисления.

Отрезок $[a, b] = [0,25; 1]$:

$$x_1 = 0,5875, \quad x_2 = 0,6625, \quad y_1 = 1,2221 < y_2 = 1,2895.$$

Отрезок $[a_1, b_1] = [a, x_2] = [0,25; 0,6625]$:

$$x_1^{(1)} = 0,4356, \quad x_2^{(1)} = 0,4769, \quad y_1^{(1)} = 1,2105 > y_2^{(1)} = 1,1953.$$

Отрезок $[a_2, b_2] = [x_1^{(1)}, x_2] = [0,4356; 0,6625]$:

$$x_1^{(2)} = 0,5377, \quad x_2^{(2)} = 0,5604, \quad y_1^{(2)} = 1,1987 < y_2^{(2)} = 1,2072.$$

Отрезок $[a_3, b_3] = [x_1^{(1)}, x_2^{(2)}] = [0,4356; 0,5604]$:

$$x_1^{(3)} = 0,4918, \quad x_2^{(3)} = 0,5043, \quad y_1^{(3)} = 1,1934 > y_2^{(3)} = 1,1932.$$

Отрезок $[a_4, b_4] = [x_1^{(3)}, x_2^{(2)}] = [0,4918; 0,5604]$.

Разность $\Delta = b_4 - a_4 = 0,0686 < \varepsilon = 0,1$. Следовательно, точкой локального минимума, найденной с заданной точностью, является $x^* \approx a_4 = 0,4918$. ■

4.0. Метод золотого сечения

Термин "золотое сечение" ввел Леонардо да Винчи. Точка x_1 является золотым сечением отрезка $[a, b]$, если отношение длины $b-a$ всего отрезка к длине $b-x_1$ большей части равно отношению длины большей части к длине x_1-a меньшей части (рис.30), т.е. x_1 - золотое сечение, если справедливо отношение
$$\frac{b-a}{b-x_1} = \frac{b-x_1}{x_1-a}.$$

Аналогично, точка x_2 , симметричная точке x_1 относительно середины отрезка $[a, b]$, является вторым золотым сечением этого отрезка. Так как точки x_1 и x_2 расположены симметрично относительно середины отрезка $[a, b]$, то можно записать

$$x_{1,2} = \frac{a+b}{2} \mp k \frac{b-a}{2}. \quad (10)$$

Учитывая, что $b-x_1 = b - \frac{a+b}{2} + k \frac{b-a}{2} = (1+k) \frac{b-a}{2}$, $x_1-a = (1-k) \frac{b-a}{2}$,

и используя определение золотого сечения, вычислим число $k > 0$:

$$\frac{b-a}{b-x_1} = \frac{b-x_1}{x_1-a} \Rightarrow \frac{2}{1+k} = \frac{1+k}{1-k} \Rightarrow k^2 + 4k - 1 = 0 \Rightarrow k = \sqrt{5} - 2.$$

Отметим свойство золотого сечения, в котором легко убедиться: пусть x_1 и x_2 - два золотых сечения отрезка $[a, b]$; тогда точка x_1 одновременно является золотым сечением отрезка $[a, x_2]$, а другая точка x_2 - золотым сечением отрезка $[x_1, b]$ (рис.30).

Теперь разберем подробнее алгоритм метода золотого сечения при нахождении последовательности вложенных отрезков $[a_i, b_i]$ ($i=0, 1, \dots, n$), сужающихся к точке x^* локального минимума унимодальной функции $f(x)$.

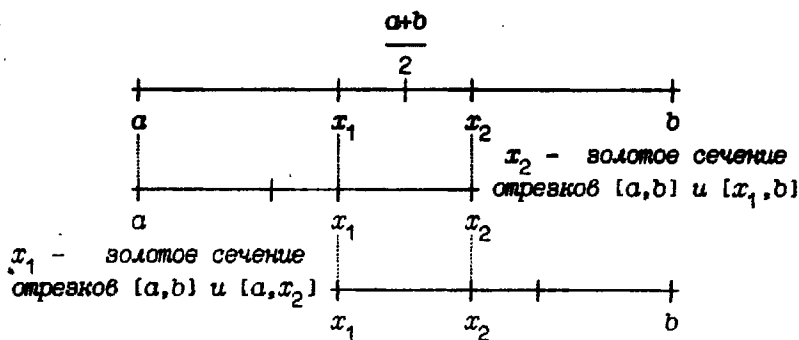


Рис.30

Сначала на исходном отрезке $[a, b]$ по формулам (10) при $k = \sqrt{5} - 2$ найдем точки x_1 и x_2 , а затем разность $\Delta_1 x = x_2 - x_1$. Далее вычисляем значения функции $y_1 = f(x_1)$ и $y_2 = f(x_2)$ и, следуя схеме п.2⁰, образуем суженный отрезок $[a_1, b_1]$. Наконец, готовясь к следующему шагу и используя свойство золотого сечения, на отрезке $[a_1, b_1]$ находим два сечения $x_1^{(1)}$, $x_2^{(1)}$. При этом возможны три случая:

$$\text{I. } y_1 < y_2, \quad a_1 = a, \quad b_1 = x_2, \quad x_2^{(1)} = x_1, \quad x_1^{(1)} = a_1 + \Delta_1 x, \quad y_2^{(1)} = y_1.$$

$$\text{II. } y_1 > y_2, \quad a_1 = x_1, \quad b_1 = b, \quad x_1^{(1)} = x_2, \quad x_2^{(1)} = b_1 - \Delta_1 x, \quad y_1^{(1)} = y_2.$$

$$\text{III. } y_1 = y_2, \quad a_1 = x_1, \quad b_1 = x_2, \quad x_{1,2}^{(1)} = \frac{a_1 + b_1}{2} \mp k \frac{b_1 - a_1}{2}.$$

Теперь, следуя разобранный схеме, можно переходить к нахождению отрезков $[a_2, b_2]$, $[a_3, b_3]$ и т. д., учитывая при этом, что в случаях I или II значение $y_2^{(t)}$ или $y_1^{(t)}$ целевой функции уже получено на предыдущем шаге ($t=1, 2, \dots, n$).

Точность приближенного равенства $x^* \sim a_n \sim b_n$ на n -м шаге вычислений можно оценить неравенством

$$0 \leq x^* - a_n \leq \frac{b-a}{\tau^n} < \varepsilon, \quad (11)$$

полученным из неравенства (8), где $\tau = \frac{2}{1+k} = \frac{2}{\sqrt{5}-1} \approx 1,618$.

Пример 4. Найти точку x^* локального минимума функции $f(x) = 2x^2 - \ln x$ на отрезке $[0,25; 1]$ методом золотого сечения с точностью $\varepsilon = 0,1$. Провести вычисления, предварительно оценив минимальное число шагов n для достижения точности ε .

Решение. Определим минимальное число шагов n , используя неравенство (11):

$$\frac{b-a}{\tau^n} < \varepsilon; \quad n > \frac{\ln[(b-a)/\varepsilon]}{\ln \tau} = \frac{\ln(0,75/0,1)}{\ln 1,618} \approx 4,187; \quad n_{\min} = 5.$$

Произведем последовательные вычисления.

Отрезок $[a, b] = [0,25; 1]$:

$$x_1 = 0,5365, \quad x_2 = 0,7135, \quad y_1 = 1,1983 < y_2 = 1,3558, \quad \Delta x_1 = 0,177.$$

Отрезок $[a_1, b_1] = [a, x_2] = [0,25; 0,7135]$:

$$x_1^{(1)} = 0,4271, \quad x_2^{(1)} = x_1, \quad y_1^{(1)} = 1,2156 > y_2^{(1)} = y_1, \quad \Delta x_2 = 0,1094.$$

Отрезок $[a_2, b_2] = [x_1^{(1)}, x_2] = [0,4271; 0,7135]$:

$$x_1^{(2)} = x_2^{(1)} = x_1, \quad x_2^{(2)} = 0,6041, \quad y_1^{(2)} = y_1 < y_2^{(2)} = 1,2339, \quad \Delta x_3 = 0,06763.$$

Отрезок $[a_3, b_3] = [x_1^{(1)}, x_2^{(2)}] = [0,4271; 0,6041]$:

$$x_1^{(3)} = 0,4947, \quad x_2^{(3)} = x_1^{(2)} = x_2^{(1)} = x_1, \quad y_1^{(3)} = 1,1933 < y_2^{(3)} = y_1, \quad \Delta x_4 = 0,0418.$$

Отрезок $[a_4, b_4] = [x_1^{(1)}, x_2^{(3)}] = [0,4271; 0,5365]$:

$$x_1^{(4)} = 0,4688, \quad x_2^{(4)} = x_1^{(3)}, \quad y_1^{(4)} = 1,1971 > y_2^{(4)} = y_1^{(3)}, \quad \Delta x_5 = 0,0258.$$

Отрезок $[a_5, b_5] = [x_1^{(4)}, x_2^{(3)}] = [0,4688; 0,5365]$.

Точкой минимума функции $f(x)$ с погрешностью $\Delta = b_5 - a_5 = 0,0677 < 0,1 = \varepsilon$ является $x^* \approx a_5 = 0,4688$. ■

3°. Метод сканирования

Пусть функция $y=f(x)$ является унимодальной на некотором промежутке. Предположим, что произвольная точка x_0 этого промежутка является исходной для поиска точки x^* локального минимума и число ϵ — заданная точность нахождения x^* . Обозначим через h произвольное приращение аргумента x и, сделав один шаг от точки x_0 , получим новое значение аргумента $x_1 = x_0 + h$.

Сравним значения функции $y_0 = f(x_0)$ и $y_1 = f(x_1) = f(x_0 + h)$. Возможны три различных продолжения в приближении к точке x^* .

I. $y_1 < y_0$ — произошло уменьшение значения функции. Тогда примем в качестве нового стартового значения $x_0^{(1)} = x_1$ и сделаем шаг h от этой точки $x_0^{(1)}$ к точке $x_1^{(1)}$, т.е. $x_1^{(1)} = x_0^{(1)} + h$. Если окажется $y_1^{(1)} < y_0^{(1)}$, то снова сделаем шаг h от новой стартовой точки $x_0^{(2)} = x_1^{(1)}$ и т.д. На некотором k -м шаге произойдет увеличение значения функции, т.е. $y_1^{(k)} > y_0^{(k)}$, и если при этом $|h| < \epsilon$, то принимаем $x^* \approx x_0^{(k)}$ с погрешностью $|h|$. В противном случае полагаем, что точка $\tilde{x}_0 = x_0^{(k)}$ является исходной для продолжения вычислений по следующей схеме II.

II. $y_1 > y_0$ — значение функции возросло. В этом случае полагаем, что начальной точкой вычислений является точка $\tilde{x}_0 = x_1$, а меньшим шагом в продолжении счета — величина $\tilde{h} = -h/K$, где K — некоторое целое положительное число, $K \geq 2$. Далее производим вычисления по схеме I или II, вплоть до достижения заданной точности.

III. $y_1 = y_0$. В этом практически маловероятном случае (опущенном при рассмотрении случаев I и II) естественно либо принять $x^* = (x_0 + x_1)/2$ при достижении заданной точности $|h| < \epsilon$, либо следовать схеме II.

Поиск минимума функции одной переменной указанным методом представляет собой колебательный процесс, совершающийся около

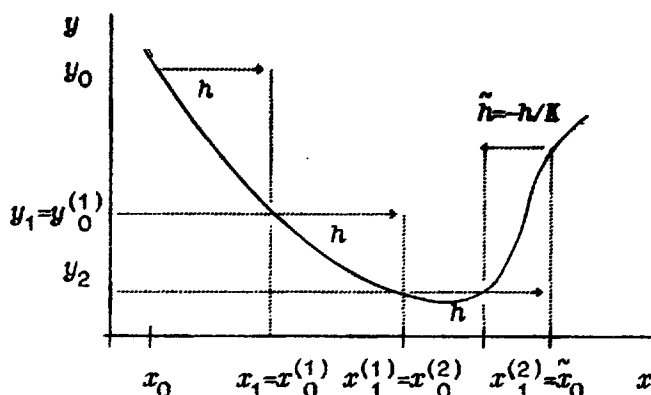


Рис.31

точки x^* локального минимума функции $f(x)$ с непрерывно уменьшающейся амплитудой (рис.31).

Описанный метод поиска локального минимума называют методом сканирования (scan - "проводить исследования по определенному плану").

Пример 5. Найти точку x^* локального минимума функции $f(x) = 2x^2 - \ln x$ методом сканирования с точностью $\varepsilon = 0,1$, полагая $x_0 = 0,25$; $h = 0,2$ и $K = 4$.

Решение. Последовательность вычислений представим в виде пар координат

ординат $\begin{bmatrix} x_t^{(k)} \\ y_t^{(k)} \end{bmatrix}$, где $t = 0; 1$, а верхний индекс k является

номером очередного шага сканирования. Имеем:

$$h = 0,2 > \varepsilon = 0,1;$$

$$x_0 = 0,25; \quad x_1 = x_0 + h = 0,45; \quad x_0^{(1)} = x_1; \quad x_1^{(1)} = 0,65;$$

$$y_0 = 1,511294 > y_1 = 1,203508; \quad y_0^{(1)} = y_1 < y_1^{(1)} = 1,275783;$$

$$\tilde{h} = -h/K = -0,05, \quad |\tilde{h}| < \varepsilon = 0,1;$$

$$\tilde{x}_0 = x_1^{(1)}; \quad \tilde{x}_1 = \tilde{x}_0 + \tilde{h} = 0,60; \quad \tilde{x}_0^{(1)} = \tilde{x}_1; \quad \tilde{x}_1^{(1)} = 0,55;$$

$$\tilde{y}_0 = y_1^{(1)} > \tilde{y}_1 = 1,230826; \quad \tilde{y}_0^{(1)} = \tilde{y}_1 > \tilde{y}_1^{(1)} = 1,202837;$$

$$\begin{aligned} \tilde{x}_0^{(2)} &= \tilde{x}_1^{(1)}; \quad \tilde{x}_1^{(2)} = \tilde{x}_0^{(2)} + \tilde{h} = 0,50; & \tilde{x}_0^{(3)} &= \tilde{x}_1^{(2)}; \quad \tilde{x}_1^{(3)} = 0,45; \\ \tilde{y}_0^{(2)} &= \tilde{y}_1^{(1)} > \tilde{y}_1^{(2)} = 1,193147; & \tilde{y}_0^{(3)} &= \tilde{y}_1^{(2)} < \tilde{y}_1^{(3)} = 1,203508. \end{aligned}$$

Здесь при вычислениях с шагом $\tilde{h}$ от точки $\tilde{x}_0 = 0,65$ до точки $\tilde{x}_1^{(2)} = 0,5$ происходило уменьшение значений целевой функции, а в точке $\tilde{x}_1^{(3)} = 0,45$ значение функции возросло. Следовательно, можно принять, что минимальное значение функции достигается в точке $x^* \approx \tilde{x}_1^{(2)} = 0,5$ с погрешностью 0,05. ■

Замечание 1. Для определения точки x^* локального минимума метод сканирования применим без предварительного нахождения промежутка унимодальности целевой функции $f(x)$. Задавая различные начальные точки x_0 и начальный шаг h , с помощью метода сканирования можно найти различные точки локального минимума, если целевая функция имеет не единственный минимум.

Замечание 2. Если предварительно получен отрезок $[a, b]$ унимодальности функции, то естественно выбирать начальную точку x_0 , принадлежащую отрезку $[a, b]$, а шаг h — таким, чтобы результаты вычислений принадлежали отрезку $[a, b]$.

Замечание 3. Для целевой функции $f(x)$, имеющей непрерывные производные до второго порядка включительно, начальное приращение аргумента h в точке x_0 можно вычислять по формуле $h = -f'(x_0) / f''(x_0)$. Действительно, запишем разложение Тейлора функции $f(x)$ в окрестности точки x_0 по степеням h , ограничиваясь квадратичными членами:

$$f(x_0+h) \approx f(x_0) + f'(x_0)h + \frac{1}{2} f''(x_0)h^2.$$

В точке минимума x^* производная по h равна нулю, т.е.

$$f'(x^*) = f'(x_0+h) = 0.$$

Таким образом, получаем уравнение $f'(x_0) + f''(x_0)h = 0$, откуда находим величину h .

Упражнения

Для заданной целевой функции $f(x)$ найти промежуток $X \subset \mathbb{R}$, на котором она унимодальна. Найти точное решение задачи одномерной минимизации $f(x) \rightarrow \min, x \in X$ ($X \subset \mathbb{R}$). Найти приближенное решение этой задачи с точностью $\varepsilon = 10^{-2}$: а) методом половинного деления; б) методом золотого сечения; в) методом сканирования.

$$1. \quad \frac{x^3}{3} + x^2. \quad 2. \quad x^3 + 6x^2 + 9x. \quad 3. \quad \frac{x^2}{x-2}.$$

$$4. \quad x^3 + \frac{x^4}{4}. \quad 5. \quad \frac{x^4}{4} - 2x^2 \quad (x > 0).$$

$$6. \quad 2x - 3\sqrt[3]{x^2}. \quad 7. \quad \frac{(x-1)^2}{x^2+1}. \quad 8. \quad x e^{-\frac{x^2}{2}}.$$

$$9. \quad x - 2\ln x. \quad 10. \quad x^{2/3}(x-5). \quad 11. \quad \frac{x^2}{2-x}.$$

$$12. \quad \frac{x^4}{4} - 2x^2 \quad (x < 0). \quad 13. \quad 1 + \sqrt[3]{(x-1)^2}. \quad 14. \quad \frac{x^2-3}{x+2}.$$

$$15. \quad 3x^4 - 8x^3 + 6x^2. \quad 16. \quad -\frac{\ln x}{x}. \quad 17. \quad \frac{3-x^2}{x+2}.$$

$$18. \quad \frac{x^2+1}{x} \quad (x > 0). \quad 19. \quad (1-x^2)(x^3-1). \quad 20. \quad \frac{(4-x)^3}{9(2-x)}.$$

$$21. \quad x\sqrt{2-x^2}. \quad 22. \quad (1-x^2)(1-x^3). \quad 23. \quad -x-2\sqrt{-x}.$$

$$24. \quad (x-2)^{2/3}(2x+1). \quad 25. \quad (x-5)e^x.$$

§4. ЧИСЛЕННЫЕ МЕТОДЫ ПОИСКА МИНИМУМА ФУНКЦИИ НЕСКОЛЬКИХ ПЕРЕМЕННЫХ

1⁰. Общая схема методов спуска

Пусть требуется решить задачу (2):

$$f(\mathbf{x}) \rightarrow \min, \mathbf{x} \in \mathbb{R}^n.$$

В двумерном пространстве $\mathbb{R}^2$ решению этой задачи можно дать геометрическую иллюстрацию. Пусть точка $\mathbf{x} = (x_1, x_2)$ лежит на плоскости Ox_1x_2 . Введем третью координату x_3 так, чтобы ось координат Ox_3 была перпендикулярна плоскости Ox_1x_2 (рис. 32). Уравнению $x_3 = f(x_1, x_2)$ соответствует поверхность в трехмерном пространстве.

Если функция $f(\mathbf{x})$ достигает локального минимума в точке $\mathbf{x}^* \in \mathbb{R}^2$, то поверхность $x_3 = f(x_1, x_2)$ в некоторой окрестности точки $\mathbf{x}^*$ имеет форму чаши (рис.32).

Напомним, что линиями уровня функции $f(x_1, x_2)$ называют семейство линий плоскости $\mathbb{R}^2$, на которых функция принимает постоянное значение. Неявным уравнением линии уровня является уравнение $f(x_1, x_2) = C$. Если функция $f(\mathbf{x})$ имеет в $\mathbb{R}^2$ единственную точку локального минимума $\mathbf{x}^*(x_1^*, x_2^*)$, то такая функция называется

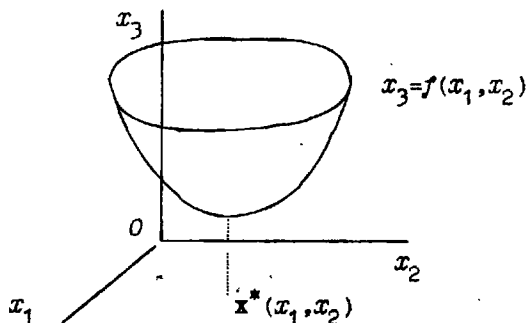


Рис.32

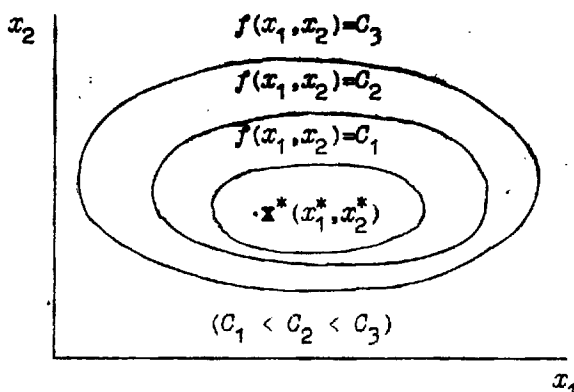


Рис.33

мономодальной. Взаимное расположение ее линий уровня имеет вид, изображенный на рис.33.

Мультимодальными называются функции, которые имеют более одного экстремума. Такова, например, функция Химмельблау

$$f(\mathbf{x}) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2,$$

имеющая четыре изолированные точки минимума. На рис.34 схематично изображены линии уровня этой функции.

Чтобы найти точку $\mathbf{x}^*$ локального минимума функции $f(\mathbf{x})$, составляют последовательность точек (приближений к решению) $\{\mathbf{x}^{(k)}\}$ ($k=0, 1, \dots$), сходящуюся к точке $\mathbf{x}^*$.

Последовательность значений функции $f(\mathbf{x}^{(k)})$ должна быть монотонно убывающей и ограниченной снизу:

$$f(\mathbf{x}^{(0)}) \geq f(\mathbf{x}^{(1)}) \geq \dots \geq f(\mathbf{x}^{(k)}) \geq \dots \geq f(\mathbf{x}^*).$$

Геометрический образ решения задачи (2) для случая двух переменных напоминает спуск на дно чаши. Это мотивирует названия методов решения задачи (2) — «методы спуска».

Для различных методов спуска сначала выбирают начальную точку последовательности $\mathbf{x}^{(0)}$. Дальнейшие приближения $\mathbf{x}^{(k)}$ опреде-

лится соотношениями

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + t^{(k)} \mathbf{S}^{(k)} \quad (k=0, 1, 2, \dots), \quad (12)$$

где $\mathbf{S}^{(k)}$ – вектор направления спуска; скалярная величина $t^{(k)}$ является решением задачи одномерной минимизации

$$f(\mathbf{x}^{(k)} + t \mathbf{S}^{(k)}) \rightarrow \min, \quad t \in \mathbb{R}. \quad (13)$$

Таким образом, задача поиска минимума функции нескольких переменных сводится к последовательности задач одномерной минимизации (13) по переменной t на отрезках n -мерного пространства, проходящих через точки $\mathbf{x}^{(k)}$ в направлении векторов $\mathbf{S}^{(k)}$.

Методы спуска различаются выбором вектора спуска и способом решения задачи одномерной минимизации.

При решении последовательности задач (12) можно ограничиться методом сканирования для поиска минимума функции одной переменной. Выбрав произвольно начальную точку $\mathbf{x}^{(0)}$ и размер начального шага h по переменной t , в методе сканирования можно по-

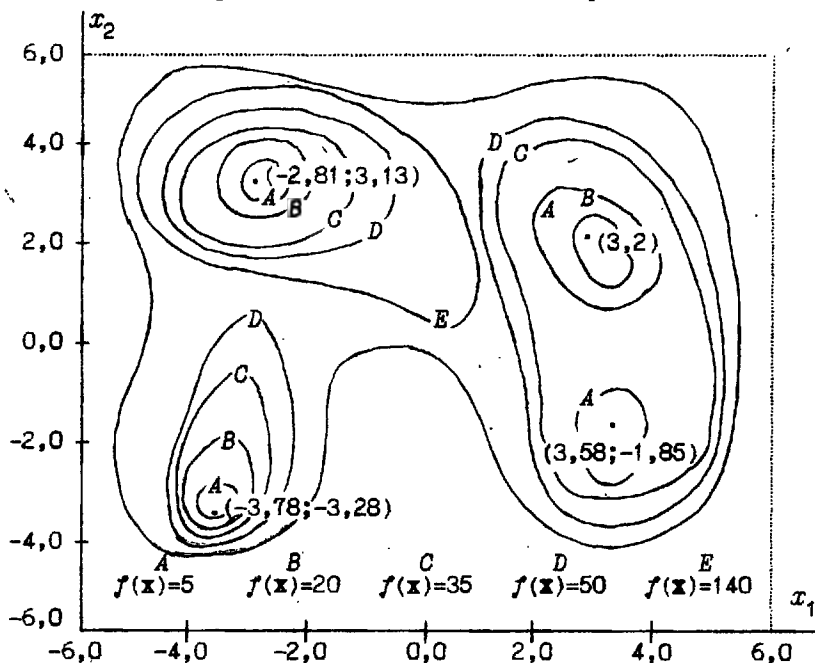


Рис. 34

лучить различные точки минимума мультимодальной функции.

Если функция $f(x)$ мономодальна, то независимо от выбора начальной точки траектория поиска должна привести к единственной точке локального минимума этой функции.

2⁰. Метод покоординатного спуска

Решение задачи (2) методом покоординатного спуска, иначе называемого методом Гаусса – Зейделя, производят по следующей общей схеме.

Выбирают произвольно начальную точку $x^{(0)}$ из области определения функции $f(x)$. Приближения $x^{(k)}$ определяются соотношениями (12):

$$x^{(k+1)} = x^{(k)} + t^{(k)} S^{(k)} \quad (k=0, 1, 2, \dots),$$

где вектор направления спуска $S^{(k)}$ — это единичный вектор, совпадающий с каким-либо координатным направлением (например, если $S^{(k)}$ параллелен x_1 , то $S^{(k)} = \{1, 0, 0, \dots, 0\}$, если он параллелен x_2 , то $S^{(k)} = \{0, 1, 0, \dots, 0\}$ и т.д.); величина $t^{(k)}$ является решением задачи одномерной минимизации (13) и может определяться, в частности, методом сканирования.

Детальная реализация общей схемы в двумерном случае R^2 дает траекторию приближения к точке x^* методом покоординатного спуска, состоящую из звеньев ломаной, соединяющих точки $x^{(k)}$, $\tilde{x}^{(k)}$, $x^{(k+1)}$ ($k=0, 1, 2, \dots$) (рис.35). При $k=0$, исходя из начальной точки $x^{(0)} = (x_1^{(0)}, x_2^{(0)})$, находят точку $\tilde{x}^{(0)} = (\tilde{x}_1^{(0)}, x_2^{(0)})$ минимума функции одной переменной $f(x_1, x_2^{(0)})$; при этом $f(\tilde{x}^{(0)}) \leq f(x^{(0)})$. Затем находят точку минимума $x^{(1)}$ функции $f(\tilde{x}_1^{(0)}, x_2)$ по второй координате. Далее делают следующий шаг вычислений при $k=1$. Полагают, что исходной точкой расчета является $x^{(1)}$. Фиксируя вторую координату точки $x^{(1)}$, находят точку минимума $\tilde{x}^{(1)} = (\tilde{x}_1^{(1)}, x_2^{(1)})$ функции $f(x_1, x_2^{(1)})$ одной переменной x_1 ; при

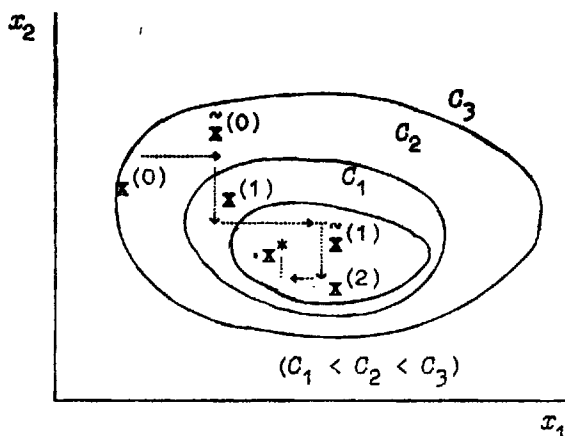


Рис.35

этом $f(\tilde{\mathbf{x}}(1)) \leq f(\mathbf{x}(1)) \leq f(\mathbf{x}(0))$. Точку $\mathbf{x}(2)$ получают, минимизируя целевую функцию $f(\tilde{x}_1^{(1)}, x_2)$ вновь по координате x_2 , фиксируя координату $\tilde{x}_1^{(1)}$ точки $\mathbf{x}(1)$, и т.д.

Условием прекращения вычислительной процедуры при достижении заданной точности ε может служить неравенство

$$\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\| < \varepsilon. \quad (14)$$

Пример 1. Методом покоординатного спуска найти точки локального минимума функции Химмельблау

$$f(\mathbf{x}) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$$

с точностью $\varepsilon = 10^{-2}$.

Решение. Решение примера рекомендуется осуществить по программе, приведенной в Приложении (см. с. 348–354). Начальный шаг h в методе сканирования при решении задачи одномерной минимизации (13) во всех приведенных вариантах принят равным 0,2.

Найдем точки локального минимума функции Химмельблау с заданной точностью, достигаемой на 6-м шаге вычислений.

Выберем следующие четыре различные начальные точки $\mathbf{x}^{(0)}$ для каждого варианта расчета: $(0,0)$, $(0,4)$, $(0,-1)$, $(0,-4)$. Координаты точек $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)})$, вычисленных на k -м шаге итерации, соот-

ответствующие значения целевой функции $f(\mathbf{x}^{(k)})$ на каждом шаге и координаты точки минимума $\mathbf{x}^* = (x_1^*, x_2^*)$ с шестью верными знаками сведены в таблицы.

Варианты расчета для начальных точек (0,0) и (0,4):

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$
0	0	0	170	0	4	130
1	3,400001	1,715625	5,609657	-2,840624	3,134376	0,041687
6	2,996876	2,000000	0,000361	-2,809374	3,128126	0,001015
	x_1^*	x_2^*	$f(\mathbf{x}^*)$	x_1^*	x_2^*	$f(\mathbf{x}^*)$
	3	2	0	-2,805118	3,131313	0

Варианты расчета для начальных точек (0,-1) и (0,-4):

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$
0	0	-1	180	0	-4	306
1	3,518751	-1,828125	0,218756	-3,949980	-3,334375	1,636352
6	3,581251	-1,850000	0,000621	-3,781248	-3,287499	0,000804
	x_1^*	x_2^*	$f(\mathbf{x}^*)$	x_1^*	x_2^*	$f(\mathbf{x}^*)$
	3,584428	-1,848126	0	-3,779310	-3,283186	0

3⁰. Метод скорейшего спуска

Пусть требуется решить задачу безусловной минимизации

$$f(\mathbf{x}) \rightarrow \min, \mathbf{x} \in \mathbb{R}^n.$$

Исходя из некоторой начальной точки $\mathbf{x}^{(0)}$, строят последовательность приближений

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + t^{(k)} \mathbf{s}^{(k)} \quad (k=0,1,2,\dots),$$

где $\mathbf{s}^{(k)}$ — единичный вектор в направлении градиента функции $f(\mathbf{x})$ в точке $\mathbf{x}^{(k)}$:

$$\mathbf{s}^{(k)} = \nabla f(\mathbf{x}^{(k)}) / \|\nabla f(\mathbf{x}^{(k)})\|.$$

Точку $\mathbf{x}^{(k+1)}$ определяют из решения задачи одномерной минимизации функции $f(\mathbf{x}^{(k)} + t\mathbf{s}^{(k)})$ по переменной t в направлении вектора $\mathbf{s}^{(k)}$:

$$f(\mathbf{x}^{(k+1)}) = f(\mathbf{x}^{(k)} + t^{(k)}\mathbf{s}^{(k)}) = \min_{t \in \mathbb{R}} f(\mathbf{x}^{(k)} + t\mathbf{s}^{(k)}). \quad (15)$$

Задачу (15) численно решают методом сканирования. Вычислительную процедуру повторяют до выполнения неравенства (14).

Напомним, что вдоль направления вектора градиента функции происходит либо наибольший рост, либо наибольшее убывание функции $f(\mathbf{x})$.

Изложенный метод поиска локального минимума функции $f(\mathbf{x})$ называется методом скорейшего или градиентного спуска. На рис.36 показана траектория поиска точки минимума в двумерном случае — ломаная с началом в точке $\mathbf{x}^{(0)}$, заканчивающаяся в точке, близкой к точке $\mathbf{x}^*$ локального минимума. Отрезок ломаной, соединяющий точки $\mathbf{x}^{(k)}$ и $\mathbf{x}^{(k+1)}$ ($k=0,1,\dots$), параллелен вектору градиента функции $f(\mathbf{x})$ в точке $\mathbf{x}^{(k)}$, сориентированного перпендикулярно линии уровня функции, проходящей через точку $\mathbf{x}^{(k)}$.

Пример 2. Методом скорейшего спуска найти точки локального минимума функции Химмельблау

$$f(\mathbf{x}) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$$

с точностью $\varepsilon = 10^{-2}$.

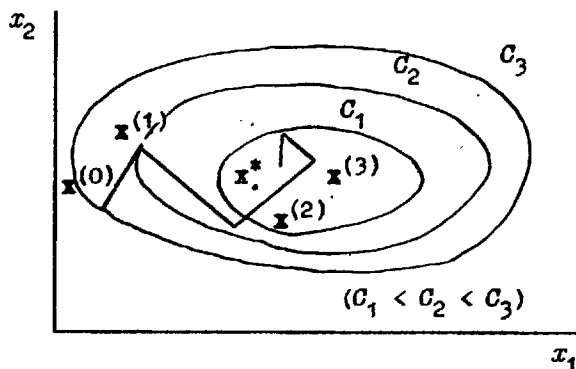


Рис.36

Решение. В расчетах по программе, приведенной в Приложении (см. с. 355–364), используются компоненты градиента функции Хаммеля:

$$\text{grad}_x f(x_1, x_2) = \frac{\partial f}{\partial x_1} = 4x_1^3 + 4x_1x_2 + 2x_2^2 - 42x_1 - 14,$$

$$\text{grad}_y f(x_1, x_2) = \frac{\partial f}{\partial x_2} = 4x_2^3 + 4x_1x_2 + 2x_1^2 - 26x_2 - 22,$$

а также компоненты единичного вектора в направлении градиента функции:

$$s_1(x_1, x_2) = \frac{\text{grad}_x f(x_1, x_2)}{\|\text{grad} f(x_1, x_2)\|}, \quad s_2(x_1, x_2) = \frac{\text{grad}_y f(x_1, x_2)}{\|\text{grad} f(x_1, x_2)\|},$$

где

$$\|\text{grad} f(x_1, x_2)\| = (\text{grad}_x^2 f(x_1, x_2) + \text{grad}_y^2 f(x_1, x_2))^{1/2}.$$

Результаты расчетов по программе для различных начальных точек $\mathbf{x}^{(0)}$ приведены в таблицах.

Варианты расчета для начальных точек (0,0) и (0,4):

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$
0	0	0	170	0	4	130
1	1,781756	2,799901	32,125792	-0,150868	2,910393	66,815132
6	3,001121	2,002225	0,000181	-2,805063	3,133298	0,000159
	x_1^*	x_2^*	$f(\mathbf{x}^*)$	x_1^*	x_2^*	$f(\mathbf{x}^*)$
	3	2	0	-2,805118	3,131313	0

Варианты расчета для начальных точек (0,-1) и (0,-4):

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$	$f(\mathbf{x}^{(k)})$
0	0	-1	180	0	-4	306
1	3,512499	-1	6,301668	-0,209979	-1,970207	178,161802
6	3,580887	-1,847277	0,000646	-3,781304	-3,274860	0,003751
	x_1^*	x_2^*	$f(\mathbf{x}^*)$	x_1^*	x_2^*	$f(\mathbf{x}^*)$
	3,584428	-1,848126	0	-3,779310	-3,283186	0

нимума целевой функции:

$$\Phi(\mathbf{x}) = f_1^2(x_1, x_2) + f_2^2(x_1, x_2) = (x_1^2 + x_2^2 - 1)^2 + (x_2 - x_1 \sin x_1)^2.$$

Упражнения

Найти приближенное решение задачи

$$f(\mathbf{x}) \rightarrow \min, \mathbf{x} \in \mathbb{R}^2,$$

где вид целевой функции $f(\mathbf{x})$ определен в упражнениях 1–25 к §2. Использовать: а) метод покоординатного спуска; б) метод скорейшего спуска.

Найти приближенное решение системы двух уравнений

$$\begin{cases} f_1(x_1, x_2) = 0, \\ f_2(x_1, x_2) = 0. \end{cases}$$

сводя эту задачу к задаче безусловной минимизации

$$\Phi(\mathbf{x}) = f_1^2(x_1, x_2) + f_2^2(x_1, x_2) \rightarrow \min, \mathbf{x} \in \mathbb{R}^2.$$

Использовать системы, заданные в упражнениях 1–25 к §4 гл. 3. Решение задачи безусловной минимизации получить с помощью метода покоординатного спуска или метода скорейшего спуска с точностью $\varepsilon = 10^{-2}$.

ПРИЛОЖЕНИЯ

В приложениях приводятся блок-схемы и тексты программ практически для всех рассмотренных численных методов на языках BASIC, PASCAL, FORTRAN и C; три программы - на языке QUICKBASIC.

ОСНОВНЫЕ БЛОКИ И ЛОГИЧЕСКИЕ УПРАВЛЕНИЕ СТРУКТУРЫ

Блок-схема представляет графическую запись алгоритма программы в виде последовательности геометрических фигур, называемых блоками. Блок отображает некоторый шаг решения задачи и содержит пояснения действий на этом шаге. Каждой форме блока соответствует определенное назначение. Блоки связаны между собой стрелками, указывающими последовательность выполнения шагов алгоритма. Некоторые определенные связи между блоками называют структурами. При построении любой программы используют последовательную структуру, условную структуру и структуру повторения. Формы блоков и правила построения схем определены стандартами.

Наиболее часто используемые блоки и их назначение

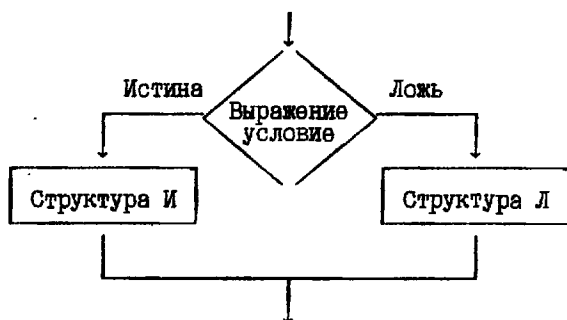
Блок	Назначение
	Начало или окончание блок-схемы алгоритма; начало или останов в программе
	Ввод исходных данных или вывод результатов программы
	Блок для размещения формул, обозначение арифметических операторов
	Блок для записи условия при ветвлении программы
	Блок для записи параметров цикла, их диапазона и шага изменения
	Блок для обозначения подпрограммы
	Указатель переноса связи между блоками

Основные управляющие структуры

Последовательная структура и ее блок-схема:



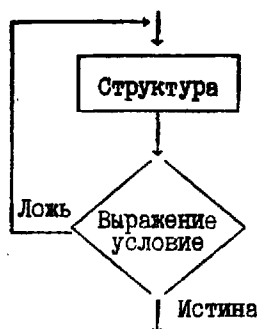
Структура условного выполнения и ее блок-схема:



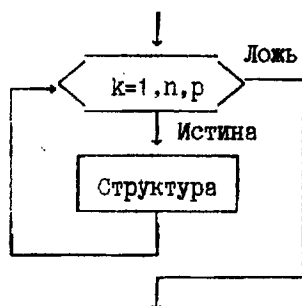
Структура повторения и ее блок-схема:



Структура повторения с постусловием и ее блок-схема:



Структура – цикл со счетчиком и ее блок-схема:

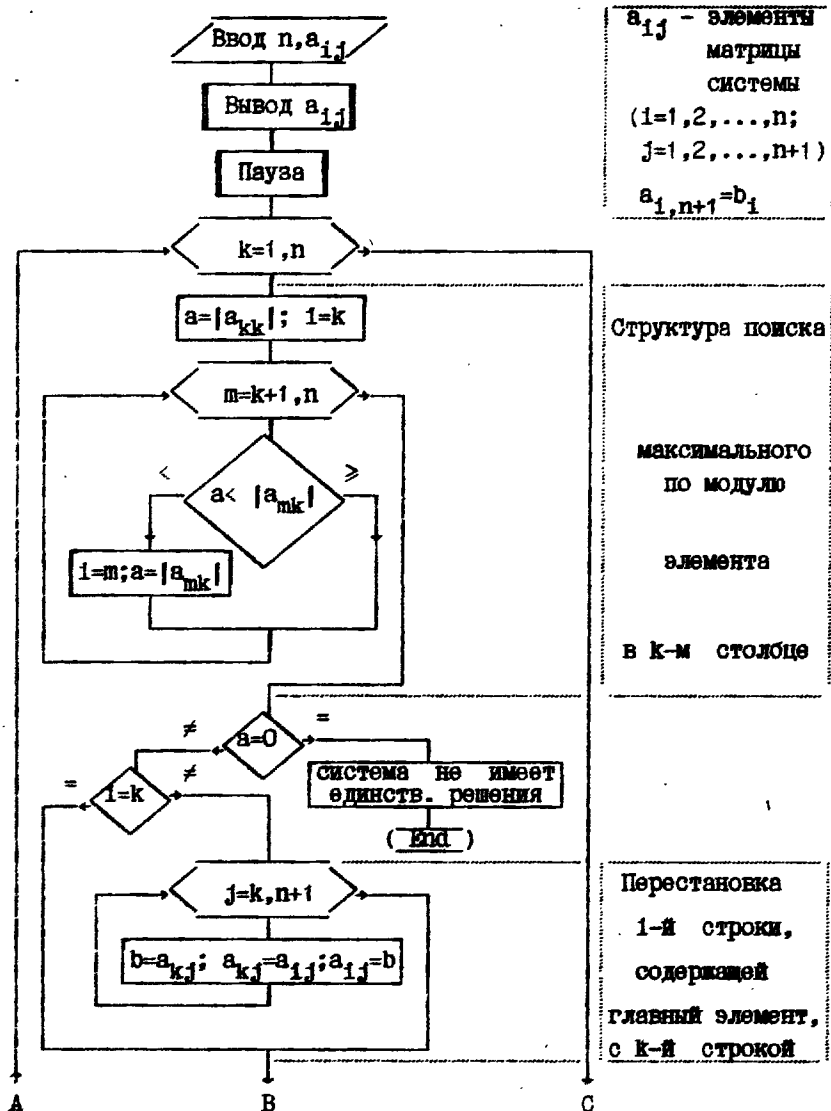


В качестве примера внутри блока цикла со счетчиком указан параметр цикла k , который изменяется от 1 до целого числа n с шагом, равным p . Выход из цикла происходит, если параметр k не принадлежит указанному диапазону изменения (на схеме обозначено как "ложь"). Можно опустить обозначение шага цикла, если $p=1$.

Цикл со счетчиком реализуется в языке BASIC в конструкции FOR...NEXT; в языке PASCAL – в виде оператора FOR с шагом цикла, равным либо 1, либо -1; в языке FORTRAN имеется оператор DO для организации цикла. В языке C организацию цикла можно осуществить, например, в виде следующей записи:

for(инициализация цикла; проверка условия выхода из цикла; коррекция параметров) структура (тело) цикла.

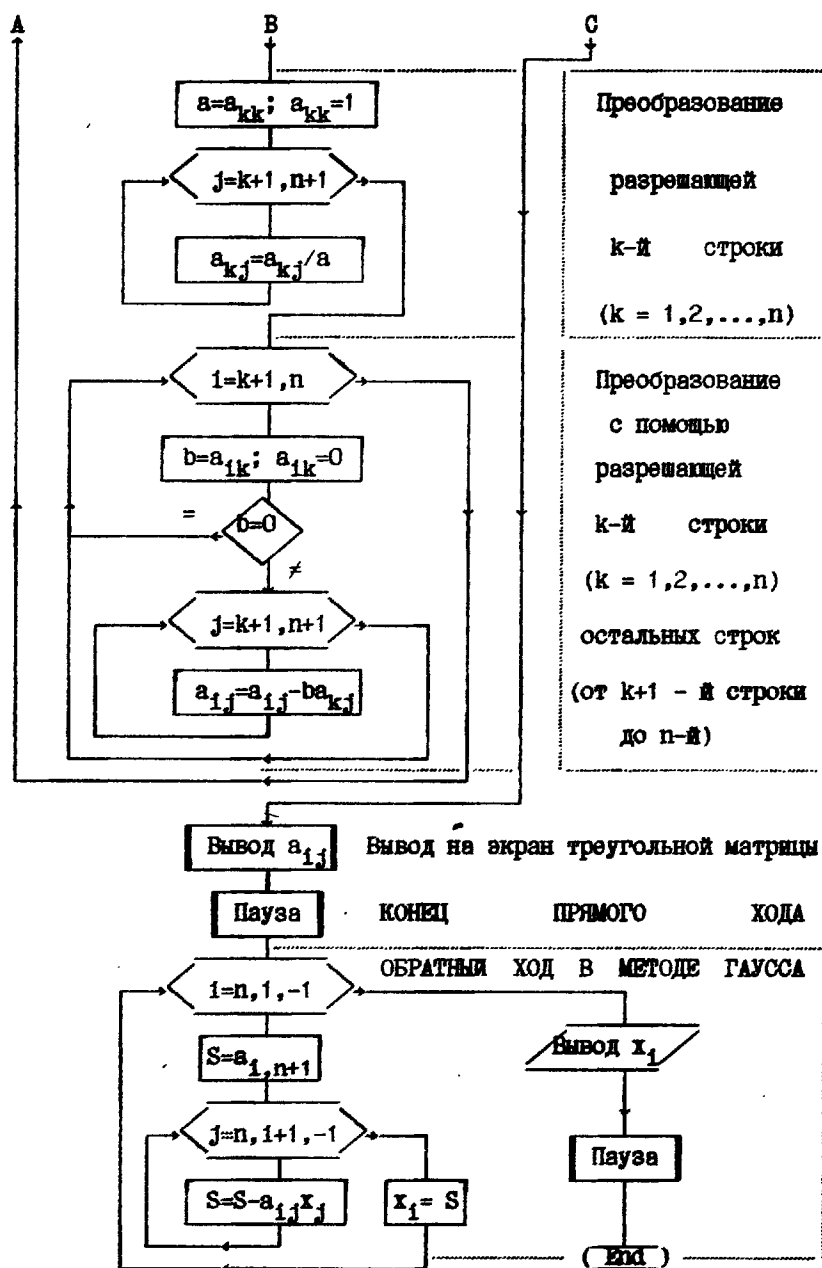
1⁰. Блок-схема решения системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента (по столбцу)



a_{ij} - элементы
матрицы
системы
($i=1, 2, \dots, n$;
 $j=1, 2, \dots, n+1$)
 $a_{i, n+1}=b_i$

Структура поиска
максимального
по модулю
элемента
в k -м столбце

Перестановка
1-й строки,
содержащей
главный элемент,
с k -й строкой



20. Программа решения системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента (по столбцу) на языке BASIC

Замечание к программам на языке BASIC. Для некоторых версий языка BASIC следует изменить разделитель ":" между операторами на разделитель "\", изменить оператор CLS - очистка экрана и символьную переменную INKEY\$, содержащую символ, считанный с клавиатуры, на соответствующий оператор и переменную в выбранной версии языка. Форматированный вывод матрицы с помощью команды PRINT USING следует изменить, например, при работе в среде TURBO BASIC. Изменяться лишь строка формата на соответствующую в рабочей версии BASIC-а.

Строки программы, не кратные 10, являются комментариями к программам или подсказками и не отражаются на решениях поставленных задач.

Примечание. Данная программа позволяет находить решение системы уравнений, если оно является единственным. Число уравнений n не должно превышать девяти для выбранного форматированного вывода расширенной матрицы системы на экран. При числе уравнений $n = 5, 6, 7, 8, 9$ каждой строке расширенной матрицы соответствуют две строки чисел на экране.

При нахождении решения системы n уравнений с n неизвестными, например для $n \leq 20$, следует переписать только строку 10 в программе на языке BASIC следующим образом:

```
10 DEFINT I-K, M-N: DIM a(20, 21), x(20)
```

```
1 REM *****
2 REM РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИИ
3 REM МЕТОДОМ ГАУССА С ВЫБОРОМ ГЛАВНОГО ЭЛЕМЕНТА
4 REM *****
10 DEFINT I-K, M-N: DIM a(9, 10), x(9)
20 CLS : INPUT "Введите число уравнений n ", n
30 PRINT "Введите расширенную матрицу по"
40 PRINT " одному элементу (по строкам)"
50 FOR i = 1 TO n
```

```

60  FOR j = 1 TO n + 1
70      INPUT ; a(1, j): PRINT "    ";
80  NEXT j: PRINT
90  NEXT i
100 GOSUB 470
101 REM -----
102 REM      Прямой      ход      в      методе      Гаусса
110  FOR k = 1 TO n
111  REM -----
112  REM      Поиск максимального по модулю элемента в k-м столбце
120      a = ABS(a(k, k)): i = k
130      FOR m = k + 1 TO n
140          IF ABS(a(m, k)) > a THEN a = ABS(a(m, k)): i = m
150      NEXT m
151  REM -----
160      IF a > .000001 THEN 180
170  PRINT "Система не имеет единственного решения": GOTO 460
171  REM -----
172  REM      Перестановка i-й строки, содержащей главный элемент,
173  REM      с k-й строкой
180      IF i = k THEN 220
190      FOR j = k TO n + 1
200          b = a(k, j): a(k, j) = a(i, j): a(i, j) = b
210      NEXT j
211  REM -----
212  REM      Преобразование k-й строки
220      a = a(k, k): a(k, k) = 1
230      FOR j = k + 1 TO n + 1
240          a(k, j) = a(k, j) / a
250      NEXT j
251  REM -----
252  REM      Преобразование с помощью k-й строки остальных строк
260      FOR i = k + 1 TO n
270          b = a(i, k): a(i, k) = 0:
280          IF b = 0 THEN 320
290          FOR j = k + 1 TO n + 1
300              a(i, j) = a(i, j) - b * a(k, j)
310          NEXT j

```



```

320 NEXT i
321 REM -----
330 NEXT k
340 GOSUB 470
341 REM Конец прямого хода в методе Гаусса
342 REM -----
343 REM          Обратный ход в методе Гаусса
350 FOR i = n TO 1 STEP -1
360   s = a(i, n + 1)
370   FOR j = n TO i + 1 STEP -1
380     s = s - a(i, j) * x(j)
390   NEXT j
400   x(i) = s
410 NEXT i: CLS
420 FOR i = 1 TO n
430   PRINT "x"; i; " = "; x(i)
440 NEXT i
450 GOSUB 530
460 END
461 REM -----
462 REM          ПОДПРОГРАММЫ
463 REM -----
470 CLS : FOR i = 1 TO n
480   FOR j = 1 TO n + 1
490     PRINT USING "    ##.###^"; a(i, j);
500   NEXT j: PRINT
510 NEXT i: GOSUB 530
520 RETURN
521 REM -----
530 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
540 IF INKEY$ = "" THEN 540
550 RETURN

```

3⁰. Программа решения системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента (по столбцу)

на языке PASCAL

```

program GAUSSmethod;
{
    *****
    РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИИ
    МЕТОДОМ ГАУССА С ВЫБОРОМ ГЛАВНОГО ЭЛЕМЕНТА
    *****
}

uses Crt;
label
    finish;
const
    l=9;
type
    matriceA=array[1..l,1..l+1] of real;
    vectorX=array[1..l] of real;
var
    i,j,k,m,n:integer;
    aa,bb,S:real;
    a:matriceA;
    x:vectorX;
    ch:char;
{
    -----
                                ПОДПРОГРАММЫ
    -----
}

procedure PAUSA;
BEGIN
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
END;
{
    -----
}

procedure OutputOfMatrice;
BEGIN
    ClrScr; FOR i:= 1 TO n DO
        BEGIN
            FOR j:= 1 TO n + 1 DO WRITE(a[i,j]:16:7);
            WRITELN;
        END;
END;

```

```

    PAUSA;
END;
{
    -----
    {
        ОСНОВНАЯ ПРОГРАММА
    }
BEGIN
    ClrScr;
    WRITELN ('Введите число уравнений n'); READ(n);
    WRITELN ('Введите расширенную матрицу по');
    WRITELN (' одному элементу (по строкам)');
    FOR i:= 1 TO n DO
        BEGIN
            FOR j:= 1 TO n + 1 DO READ(a[i, j]);
            WRITELN;
        END;
    OutputOfMatrice;
    FOR k:= 1 TO n DO
        BEGIN
            -----
            Поиск максимального по модулю элемента в k-м столбце
            aa:= ABS(a[k, k]); i:= k;
            FOR m := k+1 TO n DO
                BEGIN
                    IF ABS(a[m, k]) > aa THEN
                        BEGIN
                            i:= m; aa:= ABS(a[m, k]);
                        END;
                END;
            END;
            -----
            IF aa < 0.000001 THEN
                BEGIN
                    WRITELN;
                    WRITELN('Система не имеет единственного решения');
                    PAUSA;
                    GOTO finish;
                END;
            -----
            Перестановка i-й строки, содержащей главный элемент,
            с k-й строкой

```

```

IF 1 <> k THEN
  BEGIN
    FOR j:= k TO n + 1 DO
      BEGIN
        bb:= a[k, j]; a[k, j]:= a[1, j]; a[1, j]:= bb;
      END;
    END;
  {
    -----
    Преобразование k-й строки)
    aa:= a[k, k]; a[k, k]:= 1;
    FOR j:= k + 1 TO n + 1 DO a[k, j]:= a[k, j]/aa;
  {
    -----
    Преобразование с помощью k-й строки остальных строк)
    FOR i := k+1 TO n DO
      BEGIN
        bb:= a[i, k]; a[i,k]:=0;
        IF bb <> 0 THEN
          BEGIN
            FOR j:= k + 1 TO n + 1 DO
              a[i, j]:= a[i, j] - bb * a[k, j];
            END;
          END;
        END;
      {
        -----
      }
    END;
  OutputOfMatrice;
  {
    Конеч      прямого      хода      в      методе      Гаусса
    -----
  }
  FOR i:= n DOWNT0 1 DO
    BEGIN
      a:= a[i, n + 1];
      FOR j:= n DOWNT0 i+1 DO s:= s - a[i, j] * x[j];
      x[i]:= s;
    END;
    CLRSCR;
    FOR i:= 1 TO n DO
      WRITELN('x',i,' = ',x[i]);
    PAUSA;
  finish;; END.

```

4⁰. Программа решения системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента (по столбцу) на языке FORTRAN

Замечание к программам на языке FORTRAN. Программы написаны для версий компиляторов языка, не меньших чем 5.00, созданных фирмой Microsoft Corp.

Во всех программах для очистки экрана используется команда `cls` операционной системы. Функции DOS включаются в программу с библиотекой FORTRAN-а при компилировании программы вместе с файлом `EXEC.PI`.

Программы будут работать без очистки экрана при выводе результатов, если исключить из программ строки со следующими командами:

```
$INCLUDE: 'EXEC.PI',
INTEGER*2 system,
i=system('cls'C).
```

```
$INCLUDE: 'EXEC.PI'
```

```
PROGRAM GAUSSmethod
```

```
C * *****
C * РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ *
C * УРАВНЕНИИ МЕТОДОМ ГАУССА С ВЫБОРОМ *
C * ГЛАВНОГО ЭЛЕМЕНТА *
C * *****

INTEGER*2 system
DIMENSION x(9)
COMMON n, a(9,10)
i=system('cls'C)
WRITE (*,*) 'Введите число уравнений n'
READ (*,*) n
WRITE (*,*) 'Введите расширенную матрицу по'
WRITE (*,*) ' одному элементу (по строкам)'
DO i=1,n
DO j=1,n+1
READ (*,*) a(i,j)
END DO
```

```

WRITE(*,*)
END DO
i=system('cls'C)
CALL OutputOfMatrice
DO k=1,n

```

```

C      Поиск максимального по модулю элемента в k-м столбце
C      aa = ABS(a(k, k))
      i = k
      DO m=k+1,n
        IF(ABS(a(m,k)).GT.aa) THEN
          i = m
          aa = ABS(a(m, k))
        END IF
      END DO

```

```

C      IF (aa.LT..000001) THEN
      WRITE (*,*) ' Система не имеет единственного решения'
      PAUSE 'Нажмите клавишу ENTER для продолжения...'
      GOTO 1
      END IF

```

```

C      Перестановка 1-й строки, содержащей главный элемент,
C      с k-й строкой

```

```

      IF (i.NE.k) THEN
        DO j=k,n+1
          bb = a(k, j)
          a(k, j) = a(1, j)
          a(1, j) = bb
        END DO
      END IF

```

```

C      Преобразование          k-й          строки

```

```

      aa = a(k, k)
      a(k, k) = 1
      DO j=k+1,n+1
        a(k, j) = a(k, j) / aa
      END DO

```

```

C
C  Преобразование с помощью k-й строки остальных строк
      DO i=k+1,n
          bb = a(1, k)
          a(1, k) = 0
          IF (bb.NE.0) THEN
              DO j=k+1,n+1
                  a(1, j) = a(1, j) - bb * a(k, j)
              END DO
          END IF
      END DO

```

```

C
      END DO
      i=system('cls'C)
      CALL OutputOfMatrice
C  Конец                                прямого                                хода
C

```

```

      DO i=n,1,-1
          s = a(1, n + 1)
          DO j=n,i+1,-1
              s = s - a(1, j) * x(j)
          END DO
          x(i) = s
      END DO
      i=system('cls'C)
      DO i=1,n
          WRITE (*,'(A2,I1,A1,E12.6)') ' x',i,'=',x(i)
      END DO
      PAUSE 'Нажмите клавишу ENTER для продолжения...'
1  END
      SUBROUTINE OutputOfMatrice
          COMMON n,a(9,10)
          DO i=1,n
              WRITE (*,'(10E16.3)') (a(1,j),j=1,n+1)
          END DO
          PAUSE 'Нажмите клавишу ENTER для продолжения...'
          RETURN
      END

```

5⁰. Программа решения системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента (по столбцу) на языке C

Замечание к программам на языке C. Работа всех программ на языке C опробована в среде Turbo C++ Version 1.01 (1990 г.) фирмы Borland. Тексты программ перенесены в книгу без искажений. Для безотказной работы всех программ в этой среде должна быть установлена, в частности, опция Use C++ Compiler в позиции CPP extension only, которая находится по последовательности разворачивающихся меню Options- Compiler - C++ options. Незкономное использование массивов в программах можно оправдать только возможным облегчением при сравнении программ на языке C с программами на других языках. Блок-схемы помогают ориентироваться в структурах программ на языке C.

```
/* *****
РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ
МЕТОДОМ ГАУССА С ВЫБОРОМ ГЛАВНОГО ЭЛЕМЕНТА
***** */
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define L 9
int i,j,n;
float a[L+1][L+2];
/* -----
ПОДПРОГРАММЫ
----- */
void pausa()
{ printf("\nНажмите любую клавишу для продолжения...");
  getch();
}
/* ----- */
```



```

void outputofmatrice(void)
{ for(clrscr(),i = 1; i <= n; i++, printf("\n"))
  for(j=1; j <= n + 1; j++) printf ("%16.6f",a[i][j]);
  pausa();
}

/* -----
                   ОСНОВНАЯ      ПРОГРАММА
----- */

void main()
{ int k,m;
  float aa,bb,s,x[L+1];
  char finish;
  clrscr();
  printf ("Введите число уравнений n \n"); scanf("%i",&n);
  printf ("Введите расширенную матрицу по n");
  printf ("одному элементу (по строкам)\n");
  for (i=1; i <= n; i++, printf("\n"))
    for (j=1; j <= n + 1; scanf("%f",&a[i][j]),j++);
  for (outputofmatrice(),k=1; k<=n; k++)
  {
/* -----
    Поиск максимального по модулю элемента в k-м столбце */
    for (aa=fabs(a[k][k]), i=k, m =k+1; m<= n; m++)
      if (fabs(a[m][k]) > aa)
        { i= m; aa=fabs(a[m][k]); };
/* ----- */
    if (aa < 0.000001)
    { printf ("\nСистема не имеет единственного решения");
      pausa();
      goto finish;
    };
/* -----
    Перестановка i-й строки, содержащей главный элемент,
    с k-й строкой */
    for (j=k; (j<=n + 1) && (i != k); j++)
      { bb= a[k][j]; a[k][j]= a[i][j]; a[i][j]=bb; };
/* -----
    Преобразование k-ой строки */

```

```

for (aa=a[k][k],a[k][k]=1,j=k+1; j<=n+1; j++)
  a[k][j]= a[k][j]/aa;
/*
Преобразование с помощью k-й строки остальных строк */
for (i=k+1; i<=n; i++)
  for (bb=a[i][k],a[i][k]=0,j=k+1;(j<=n+1)&&(bb!=0);j++)
    a[i][j]=a[i][j] - bb * a[k][j];
);
/* КОНЕЦ ПРЯМОГО ХОДА В МЕТОДЕ ГАУССА
----- */
for (outputofmatrice(),i=n; i>=1; x[i]=s,i--)
for (s=a[i][n+1],j=n;j>=1+1;j--)  s = s - a[i][j]*x[j];
for (clrscr(),i=1;i<=n;i++) printf ("x%d=%f\n",i,x[i]);
pause();
finish;;
}

```

**6⁰. Программа для решения системы линейных
алгебраических уравнений методом Хордана - Гаусса
на языке QUICKBASIC**

Замечание к программам на языке QUICKBASIC. Язык QUICKBASIC (или QBASIC) - одна из версий BASIC-а фирмы Microsoft - наиболее близок к современным языкам программирования и получает широкое распространение в практике благодаря следующим достоинствам:

- чрезвычайно удобной среде с развитой системой помощи пользователю и многочисленными примерами;
- необязательным проставлением меток перед операторами;
- наличием имен подпрограмм и функций;
- возможностью пользователя определять структуры данных;
- наличием широкой библиотеки графических программ;
- возможностью формирования исполняемого файла программы (файла с расширением exe) и т.п.

Данная программа позволяет находить решение системы n уравнений с n неизвестными для любого числа n , допускаемого памятью персонального компьютера. Для числа $n > 4$ лучше исключить вывод на экран результатов промежуточных преобразований расширенной матрицы системы (исключая команды "CALL matriceout(n)" из программы)

Заметим, что среда QUICKBASIC формирует следующие команды:

```
DECLARE SUB matriceout (n%)
```

```
DECLARE SUB pausa ()
```

```
DEFBNG A-B, D, I-K, M-N
```

```
DEFSNG A, I-J, N
```

```
* *****
* РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ
* МЕТОДОМ ЖОРДАНА - ГАУССА
* С ВЫБОРОМ МАКСИМАЛЬНОГО ЭЛЕМЕНТА
* ВЫЧИСЛЕНИЕ ОПРЕДЕЛИТЕЛЯ МАТРИЦЫ СИСТЕМЫ
* *****
```

```
DECLARE SUB matriceout (n%)
```

```
DECLARE SUB pausa ()
```

```
DEFINT I-K, M-N: DEFDBL A-B, D
```

```
CLS : INPUT "Введите число уравнений n ", n
```

```
DIM a(n, n + 1), j1(n), x(n)
```

```
PRINT "Введите расширенную матрицу по"
```

```
PRINT "одному элементу (по строкам)"
```

```
FOR i = 1 TO n
```

```
    j1(i) = 1
```

```
    FOR j = 1 TO n + 1
```

```
        INPUT ; a(i, j): PRINT "    ";
```

```
    NEXT: PRINT
```

```
NEXT
```

```
det = 1
```

```
МЕТОД ЖОРДАНА - ГАУССА
```

```
FOR k = 1 TO n
```

CALL matriceout(n)

Поиск максимального по модулю элемента
в матрице a(1,j)

aa = ABS(a(k, k)): i = k: j0 = k

FOR m = k TO n

FOR p = k TO n

IF ABS(a(m, p)) > aa THEN

aa = ABS(a(m, p)): i = m: j0 = p

END IF

NEXT

NEXT

det = det * a(1, j0)

IF aa < .0000000000000001# THEN

PRINT "Система не имеет единственного решения"

PRINT "Определитель матрицы det =";0;" (" ; det;")"

CALL pausa: END

END IF

Перестановка i-й строки, содержащей максимальный
элемент, с k-й строкой

IF i <> k THEN

det = -det

FOR j = k TO n + 1

SWAP a(k, j), a(i, j)

NEXT

END IF

Перестановка j-го столбца, содержащего максимальный
элемент с k-м столбцом

IF j0 <> k THEN

det = -det

SWAP j1(k), j1(j0)

FOR i = 1 TO n

SWAP a(i, k), a(i, j0)

NEXT

END IF

' Преобразование k-й строки

a = a(k, k): a(k, k) = 1

FOR j = k + 1 TO n + 1

a(k, j) = a(k, j) / a

NEXT

' Преобразование с помощью k-й строки остальных
' строк

FOR i = 1 TO n

IF i <> k THEN

b = a(i, k): a(i, k) = 0:

FOR j = k + 1 TO n + 1

a(i, j) = a(i, j) - b * a(k, j)

NEXT

END IF

NEXT

NEXT

CALL matriceout(n)

'пузырьковая сортировка результатов вычислений

FOR i = 2 TO n

FOR j = n TO i STEP -1

IF j1(j - 1) > j1(j) THEN

SWAP j1(j - 1), j1(j)

SWAP a(j - 1, n + 1), a(j, n + 1)

END IF

NEXT

NEXT

'Вывод решения системы уравнений на экран

CLS

FOR i = 1 TO n

PRINT "x"; i; " = "; a(i, n + 1)

NEXT

PRINT "Определитель матрицы det ="; det

CALL pausa

END

```
DEFSNG A-B, D, I-K, M-N
```

```
SUB matriceout (n%)
```

```
DEFDEL A
```

```
SHARED a()
```

```
DEFINT I-J, N
```

```
CLS
```

```
FOR i = 1 TO n
```

```
FOR j = 1 TO n + 1
```

```
PRINT USING "      ##.###^ ^ ^ ^"; a(i, j);
```

```
NEXT: PRINT
```

```
NEXT
```

```
CALL pausa
```

```
END SUB
```

```
DEFSNG A, I-J, N
```

```
SUB pausa
```

```
PRINT : PRINT "Для продолжения нажмите любую клавишу"
```

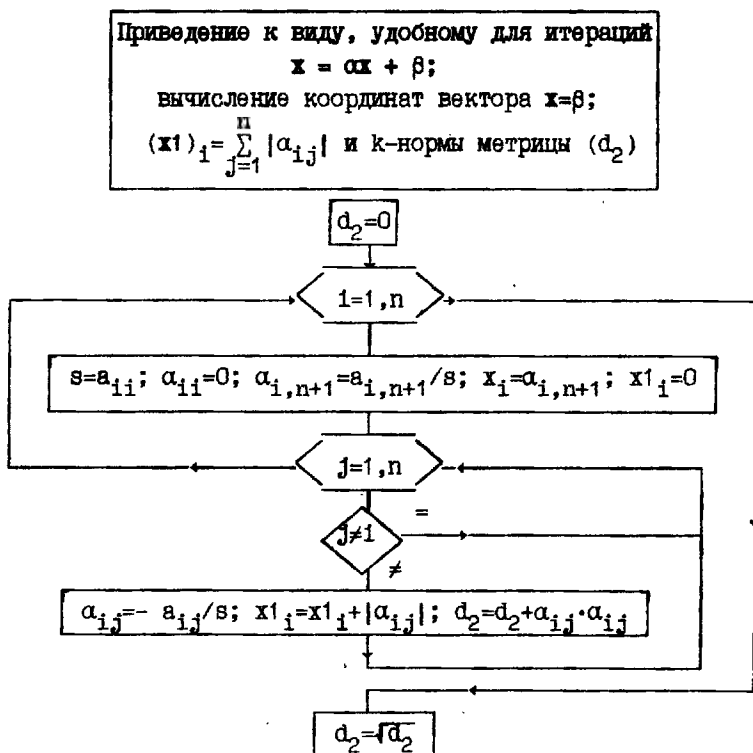
```
DO
```

```
  a$ = INKEY$
```

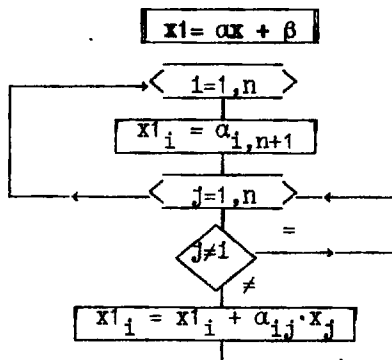
```
LOOP WHILE a$ = ""
```

```
END SUB
```


*Некоторые более подробные фрагменты
блок-схем численного решения системы уравнений методом итераций*



*Подпрограмма вычисления вектора на одном шаге итерации
(умножение матрицы α на вектор $\mathbf{x}$ плюс вектор β)*



2⁰. Программа численного решения системы
линейных алгебраических уравнений
методом итераций на языке BASIC

```

1  REM *****
2  REM                РЕШЕНИЕ                СИСТЕМЫ
3  REM  ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИИ МЕТОДОМ ИТЕРАЦИИ
4  REM  *****
10  DEFINT I-J, N: DIM a(9, 10), x(9), x1(9)
20  CLS
30  PRINT "      Введите через запятую"
40  PRINT "число уравнений n и точность epsilon"
50  INPUT n, e
60  PRINT "Введите расширенную матрицу по"
70  PRINT "одному элементу (по строкам)"
80  FOR i = 1 TO n
90    FOR j = 1 TO n + 1
100     INPUT ; a(i, j): PRINT "    ";
110    NEXT j: PRINT
120  NEXT i: GOSUB 520
121  REM -----
122  REM Проверка условия, допускающего приведение системы к виду,
123  REM                удобному для итераций
130  FOR i = 1 TO n
140    IF a(i, 1) <> 0 THEN 170
150  PRINT : PRINT "      Система решается другим методом "
160  GOSUB 580: GOTO 510
170  NEXT i
171  REM -----
172  REM Преобразование матрицы к виду, удобному для итераций;
173  REM вычисление компонент вектора x1, равных сумме по j
174  REM abs(a(i,j)); вычисление k-нормы приведенной матрицы
180  d2 = 0
190  FOR i = 1 TO n
200    s = a(i, 1): a(i, 1) = 0: a(i, n + 1) = a(i, n + 1) / s
210    x(1) = a(i, n + 1): x1(1) = 0

```

```

220   FOR j = 1 TO n
230     IF j = 1 THEN 270
240     a(1, j) = -a(1, j) / s: x1(1) = x1(1) + ABS(a(1, j))
250     d2 = d2 + ABS(a(1, j)) * ABS(a(1, j))
270   NEXT j
280 NEXT 1
290 d2 = SQR(d2)
300 GOSUB 520
-----
301 REM -----
302 REM      Вычисление m-нормы "приведенной" матрицы
310 GOSUB 830: d = y
311 REM -----
312 REM      Вычисление l-нормы "приведенной" матрицы
320 FOR i = 1 TO n
330   x1(i) = 0
340   FOR j = 1 TO n
350     IF j <> i THEN x1(i) = x1(i) + ABS(a(j, i))
360   NEXT j
370 NEXT i
380 GOSUB 830: d1 = y
390 PRINT : PRINT USING "Нормы =({#.##; #.##; #.##})"; d; d1; d2
391 REM -----
392 REM      Нахождение наименьшей из трех норм
400 IF d1 < d THEN d = d1
410 IF d2 < d THEN d = d2
420 PRINT "Наименьшая норма = "; d
421 REM -----
422 REM Проверка достаточного условия сходимости метода
430 IF d < 1 THEN 470
440 PRINT "Достаточные условия сходимости метода итераций "
450 PRINT "      не удовлетворяются"
460 GOSUB 580: GOTO 510
461 REM -----
470 PRINT "Условие сходимости метода удовлетворено": GOSUB 580
480 e = d / (1 - d)
481 REM -----
482 REM      Выполнение последовательных итераций
490 GOSUB 610: GOSUB 680: GOSUB 730: GOSUB 580

```

```

500 IF d * s > e THEN GOSUB 790: GOTO 490
510 END
511 REM -----
512 REM                                ПОДПРОГРАММЫ
513 REM -----
514 REM Вывод матрицы на экран (не более 5 чисел в строке)
520 CLS : FOR i = 1 TO n
530     FOR j = 1 TO n + 1
540         PRINT USING "      ##.###^ ^ ^ ^"; a(i, j);
550     NEXT j: PRINT
560 NEXT i: GOSUB 580
570 RETURN
571 REM -----
572 REM                                Пауза
580 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
590 IF INKEY$ = "" THEN 590
600 RETURN
601 REM -----
602 REM Вычисление вектора на каждом шаге итераций (умножение
603 REM матрицы на вектор плюс вектор свободных членов)
610 FOR i = 1 TO n
620     x1(i) = a(i, n + 1)
630     FOR j = 1 TO n
640         IF j <> i THEN x1(i) = x1(i) + a(i, j) * x(j)
650     NEXT j
660 NEXT i
670 RETURN
671 REM -----
672 REM Вычисление расстояния между векторами двух
673 REM последовательных приближений по m-норме
680 d = ABS(x(1) - x1(1))
690 FOR i = 2 TO n
700     IF ABS(x(i) - x1(i)) > d THEN d = ABS(x(i) - x1(i))
710 NEXT i
720 RETURN
721 REM -----
722 REM Вывод на экран координат вектора итерационного процесса
730 CLS

```

```

740 FOR i = 1 TO n
750   PRINT "x"; i; " = "; x1(i)
760 NEXT i
770 PRINT "Расстояние между итерациями по m-норме = "; d
780 RETURN
781 REM -----
790 FOR i = 1 TO n
800   x(i) = x1(i)
810 NEXT i
820 RETURN
821 REM -----
822 REM           Вычисление 1- m- нормы матрицы
830 y = ABS(x1(1))
840   FOR i = 2 TO n
850     IF ABS(x1(i)) > y THEN y = ABS(x1(i))
860   NEXT i
870 RETURN

```

**3⁰. Программа численного решения системы
линейных алгебраических уравнений
методом итераций на языке PASCAL.**

```

program SolutionOfSystemByIteration;
{ *****
  РЕШЕНИЕ          СИСТЕМЫ
  ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИИ МЕТОДОМ ИТЕРАЦИИ
  ***** }
uses Crt;
label
  finish;
const
  l=9;
type
  matriceA=array[1..l,1..l+1] of real;
  vectorX=array[1..l] of real;
var

```

```

1,j,n:integer;
d,d1,d2,e,s:real;
a:matriceA;
x,x1:vectorX;
ch:char;

```

```

{

```

ПОДПРОГРАММЫ

```

procedure PAUSA;

```

```

BEGIN

```

```

    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...')

```

```

    REPEAT ch := readkey UNTIL ch <> '';

```

```

END;

```

```

{

```

Вывод матрицы на экран

```

}
```

```

procedure OutputOfMatrice;

```

```

BEGIN

```

```

    ClrScr; FOR i:= 1 TO n DO

```

```

        BEGIN

```

```

            FOR j:= 1 TO n + 1 DO

```

```

                WRITE(a[i,j]:16:7);

```

```

            WRITELN;

```

```

        END;

```

```

    PAUSA;

```

```

END;

```

```

{

```

Вычисление вектора на каждом шаге итераций (умножение

матрицы на вектор плюс вектор свободных членов) }

```

procedure VectorOfIteration;

```

```

BEGIN

```

```

    FOR i := 1 TO n DO

```

```

        BEGIN

```

```

            x1[i] := a[i, n + 1];

```

```

            FOR j := 1 TO n DO

```

```

                IF j <> i THEN x1[i] := x1[i] + a[i, j] * x[j];

```

```

            END;

```

```

END;

```

```

{

```

Вычисление m - и l - норм вектора

function normlm (x: vectorX): real;

var

y: real;

BEGIN

y := ABS(x[1]);

FOR i := 1 TO n DO

if ABS(x[i]) > y then y := ABS(x[i]);

normlm := y;

END;

{ ----- }

Вывод на экран координат вектора итерационного процесса }

procedure OutputOfVector;

BEGIN

ClrScr;

FOR i := 1 TO n DO WRITELN ('x', i, '= ', x[i]);

WRITELN ('Расстояние между итерациями по m -норме = ', d);

END;

{ ----- }

procedure ChangeXbyX1;

BEGIN

FOR i := 1 TO n DO x[i] := x1[i];

END;

{ ----- }

ОСНОВНАЯ ПРОГРАММА

BEGIN

ClrScr;

WRITELN (' Введите ');

WRITELN ('число уравнений n и точность epsilon'); READ(n,e);

WRITELN ('Введите расширенную матрицу по');

WRITELN (' одному элементу (по строкам)');

FOR i:= 1 TO n DO

BEGIN

FOR j:= 1 TO n + 1 DO

READ (a[i, j]);

WRITELN;

END;

OutputOfMatrice;

```

( -----
Проверка условия - диагональные элементы отличны от 0)
  FOR i := 1 TO n DO
    BEGIN
      IF a[i, i] = 0 THEN
        BEGIN
          WRITELN;
          WRITELN (' Система решается другим методом');
          PAUSE; GOTO finish;
        END;
      END;
    END;
( Преобразование матрицы к виду, удобному для итераций; вы-
числение компонент вектора X1, равных сумме abs(a(i,j))
по j; вычисление k-нормы приведенной матрицы )
  d2 := 0;
  FOR i := 1 TO n DO
    BEGIN
      s := a[i, 1]; a[i, 1] := 0; x1[i] := 0;
      a[i, n + 1] := a[i, n + 1] / s; x1[i] := a[i, n + 1];
      FOR j := 1 TO n DO
        IF j <> 1 THEN
          BEGIN
            a[i, j] := -a[i, j] / s;
            x1[i] := x1[i] + ABS(a[i, j]);
            d2 := d2 + ABS(a[i, j]) * ABS(a[i, j]);
          END;
        END;
      d2 := SQRT(d2);
      OutputOfMatrice;
    END;
( -----
Вычисление m-нормы "приведенной " матрицы )
  d := normlm(x1);
( -----
Вычисление l-нормы "приведенной" матрицы )
  FOR i := 1 TO n DO
    BEGIN
      x1[i] := 0;
      FOR j := 1 TO n DO

```

```

        IF j <> 1 THEN x1[i] := x1[i] + ABS(a[j, i]);
    END;
d1 := normlm(x1);
WRITELN;
WRITELN ('Нормы = (' , d:4:2, ',' , d1:4:2, ',' , d2:4:2, ')');
{
    -----
    Нахождение наименьшей из трех норм
    IF d1 < d THEN d := d1;
    IF d2 < d THEN d := d2;
    WRITELN ('Наименьшая норма = ' , d:4:2);
    -----
    Проверка достаточного условия сходимости метода
    IF d >= 1 THEN
        BEGIN
            WRITELN ('Достаточные условия сходимости метода итераций');
            WRITELN ('          не удовлетворяются');
            PAUSA; GOTO finish;
        END;
    -----
    WRITELN ('Условие сходимости метода удовлетворено');
    PAUSA;
    s := d / (1 - d);
    -----
    Выполнение последовательных итераций
    -----
    REPEAT
        VectorOfIteration;
        FOR i :=1 TO n DO
            x[i] := x1[i] - x[i]; d := normlm(x);
        OutputOfVector; PAUSA;
        ChangeXbyX1;
    UNTIL d*s < e;
finish: END.

```


4⁰. Программа численного решения системы линейных алгебраических уравнений методом итераций на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
      PROGRAM SolutionOfSystemByIteration
C      *****
C      *РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ*
C      *УРАВНЕНИИ МЕТОДОМ ИТЕРАЦИИ*
C      *****
      INTEGER*2 system
      DIMENSION x(9),x1(9)
      COMMON n,a(9,10)
      i=system('cls'C)
      WRITE (*,*) '          Введите '
      WRITE (*,*) 'число уравнений n и точность epsilon'
      READ (*,*) n, e
      WRITE (*,*) 'Введите расширенную матрицу по'
      WRITE (*,*) ' одному элементу (по строкам)'
      DO i=1,n
        DO j=1,n+1
          READ (*,*) a(i,j)
        END DO
        WRITE(*,*)
      END DO
      i=system('cls'C)
      CALL OutputOfMatrice
C      -----
C      Проверка условия - диагональные элементы отличны от 0
      DO i=1,n
        IF (a(i, 1).EQ.0) THEN
          WRITE (*,*)
          WRITE (*,*) '          Система решается другим методом'
          PAUSE 'Нажмите клавишу ENTER для продолжения...'
          GOTO 1
        END IF
      END DO

```

END DO

```

C -----
C Преобразование матрицы к виду, удобному для итераций; вы-
C числение компонент вектора X1, равных сумме abs(a(1,j))
C по j; вычисление k-нормы приведенной матрицы
  d2 = 0
  DO i=1,n
    s = a(1, 1)
    a(1, 1) = 0
    x1(1) = 0
    a(1, n + 1) = a(1, n + 1) / s
    x(1) = a(1, n + 1)
  DO j=1,n
    IF (j.NE.1) THEN
      a(1, j) = -a(1, j) / s
      x1(1) = x1(1) + ABS(a(1,j))
      d2 = d2 + ABS(a(1,j))*ABS(a(1,j))
    END IF
  END DO
END DO
d2 = SQRT(d2)
i=system('cls'C)
CALL OutputOfMatrice

```

```

C -----
C      Вычисление m-нормы "приведенной " матрицы
C d = dnormlm(x1,n)

```

```

C -----
C      Вычисление l-нормы "приведенной" матрицы
C DO i = 1,n
C   x1(1) = 0
C   DO j = 1,n
C     IF (j.NE.1) x1(1) = x1(1) + ABS(a(j, 1))
C   END DO
C END DO
C d1 = dnormlm(x1,n)
C WRITE (*,*)
C WRITE(*, '(A9.3(F4.2,A1))') ' Нормы =( ' ,d, ';' ,d1, ';' ,d2, ' ) '

```

C -----

C Нахождение наименьшей из трех норм

IF (d1.LT.d) d = d1

IF (d2.LT.d) d = d2

WRITE (*, '(A19,F5.3)') ' Наименьшая норма = ', d

WRITE (*,*)

C

C Проверка достаточного условия сходимости метода

IF (d.GE.1) THEN

WRITE (*,*) 'Достаточные условия сходимости метода'

WRITE (*,*) ' итераций не удовлетворяются'

PAUSE 'Нажмите клавишу ENTER для продолжения...'

GOTO 1

END IF

C

WRITE (*,*) 'Условие сходимости метода удовлетворено'

PAUSE 'Нажмите клавишу ENTER для продолжения...'

s = d / (1 - d)

C

C Выполнение последовательных итераций

C

DO WHILE (d*s.GT.e)

CALL VectorOfIteration(x,x1)

DO i = 1,n

x(i) = x1(i) - x(i)

END DO

d = dnormlm(x,n)

i=system('cls'C)

CALL OutputOfVector(x1,n,d)

PAUSE 'Нажмите клавишу ENTER для продолжения...'

CALL ChangeXbyX1(x,x1,n)

END DO

1 END

C

C

ПОДПРОГРАММЫ

C

C

Вывод матрицы на экран

SUBROUTINE OutputOfMatrice

COMMON n,a(9,10)

```

DO i=1,n
  WRITE (*,'(10E16.3)') (a(i,j),j=1,n+1)
END DO
PAUSE 'Нажмите клавишу ENTER для продолжения...'
RETURN
END

```

```

C
C      Вычисление вектора на каждом шаге итераций (умножение
C      матрицы на вектор плюс вектор свободных членов)
SUBROUTINE VectorOfIteration(x,x1)
  DIMENSION x(9),x1(9)
  COMMON n,a(9,10)
  DO i = 1,n
    x1(i) = a(i, n + 1)
    DO j = 1,n
      IF (j.NE.1) x1(i) = x1(i) + a(i, j) * x(j)
    END DO
  END DO
  RETURN
END

```

```

C
C      Вычисление m- и l- норм вектора
function dnormlm(x,n)
  DIMENSION x(9)
  y = ABS(x(1))
  DO i = 2,n
    IF (ABS(x(i)).GT.y) y = ABS(x(i))
  END DO
  dnormlm = y
  RETURN
END

```

```

C
C      Вывод на экран координат вектора итерационного процесса
SUBROUTINE OutputOfVector(x1,n,d)
  DIMENSION x1(9)
  DO i = 1,n
    WRITE (*,'(A2,I1,A1,E12.6)') ' x',i,'=',x1(i)
  END DO

```

```

WRITE (*,*) 'Расстояние между итерациями'
WRITE (*,'(A13,E12.6)') ' по m-норме =',d
RETURN
END

```

0

```

SUBROUTINE ChangeXbyX1 (x,x1,n)
DIMENSION x(9),x1(9)
DO 1 = 1,n
    x(1) = x1(1)
END DO
RETURN
END

```

5⁰. Программа численного решения системы линейных алгебраических уравнений методом итераций на языке C

```

/*****
                                РЕШЕНИЕ      СИСТЕМЫ
ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИИ МЕТОДОМ ИТЕРАЦИИ
*****/
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define L 9
int i,j,n;
float d,x[L+1],x1[L+1],a[L+1][L+2];
/*
                                ПОДПРОГРАММЫ
                                -----*/
void pausa()
{ printf("\nНажмите любую клавишу для продолжения...");
  getch();
}
/*

```

Вывод матрицы на экран

*/

```
void outputofmatrice(void)
{for(clrscr(),i = 1; i <= n; i++,printf("\n"))
  for(j=1; j <= n + 1; j++) printf ("%16.6f",a[i][j]);
  pause();
}
```

```
/* -----
Вычисление вектора на каждом шаге итераций (умножение
матрицы на вектор плюс вектор свободных членов) */
```

```
void vectorofiteration(void)
{ for (i = 1; i <= n; i++)
  for (x[i] = a[i][n + 1], j = 1; j <= n; j++)
    if (j != 1) x[i] = x[i] + a[i][j] * x[j];
}
```

```
/* -----
Вычисление m- 1- нормы вектора */
```

```
float norm1m (float x[L+1])
{ float y;
  for (y = fabs(x[1]), i = 1; i <= n; i++)
    if (fabs(x[i]) > y) y = fabs(x[i]);
  return y;
}
```

```
/* -----
Вывод на экран координат вектора итерационного процесса*/
```

```
void outputofvector(void)
{for(clrscr(),i=1;i<=n; printf("x%d=%f\n",i,x[i]),i++);
printf("Расстояние между итерациями по m-норме = %f",d);
}
```

```
/* -----*/
void changexbyx1(void)
```

```
{ for (i = 1; i <= n; x[i] = x[i],i++); }
```

```
/* -----
ОСНОВНАЯ ПРОГРАММА
-----*/
```

```
void main()
{ float d1,d2,e,s;
  char finish;
  clrscr();
```

```

printf ("          Введите n");
printf ("число уравнений n и точность epsilon\n");
scanf ("%i%f",&n,&e);
printf ("Введите расширенную матрицу по n");
printf ("одному элементу (по строкам)\n");
    for (i=1; i <= n; i++, printf("\n"))
        for (j=1; j <= n + 1; scanf("%f",&a[i][j]),j++);
    outputofmatrice();

/* -----
Проверка условия - диагональные элементы отличны от 0 */
    for (i = 1; i <= n; i++)
        if (!a[i][i])
            { printf ("\n\n Система решается другим методом");
              pausa(); goto finish; };

/* Преобразование матрицы к виду, удобному для итераций;
вычисление компонент вектора X1, равных сумме
fabs(a(i,j)) по j; вычисление k-нормы приведенной
матрицы */
    for (d2 = 0, i = 1; i <= n; i++)
        { s = a[i][1]; a[i][1] = x1[1] = 0;
          a[i][n + 1] = a[i][n + 1] / s; x1[1] = a[i][n+1];
          for (j = 1; j <= n; j++)
              { if (j != 1)
                  { a[i][j] = -a[i][j] / s;
                    x1[1] = x1[1] + fabs(a[i][j]);
                    d2 = d2 + fabs(a[i][j]) * fabs(a[i][j]);};
              };
          d2 = sqrt(d2);
          outputofmatrice();

/* -----
          Вычисление m-нормы "приведенной" матрицы */
          d = normlm(x1);

/* -----
          Вычисление l-нормы "приведенной" матрицы */
          for (i = 1; i <= n; i++)
              for (x1[1] = 0, j = 1; j <= n; j++)
                  if (j != 1) x1[1] = x1[1] + fabs(a[j][i]);

```

```

d1 = normlm(x1);
printf ("\n\nНормы = d (%f %f %f)", d, d1, d2);

/* ----- */
    Нахождение наименьшей из трех норм                */
if (d1 < d) d = d1;
if (d2 < d) d = d2;
printf ("\n\nНаименьшая норма = %f", d);

/* ----- */
    Проверка достаточного условия сходимости метода    */
    if (d >= 1)
    {
printf ("\n Достаточные условия сходимости ");
printf ("\nметода итераций не удовлетворяются");
    pausa(); goto finish;
    };

/* ----- */
printf ("\n\nУсловие сходимости метода удовлетворено");
pausa();
s = d / (1 - d);

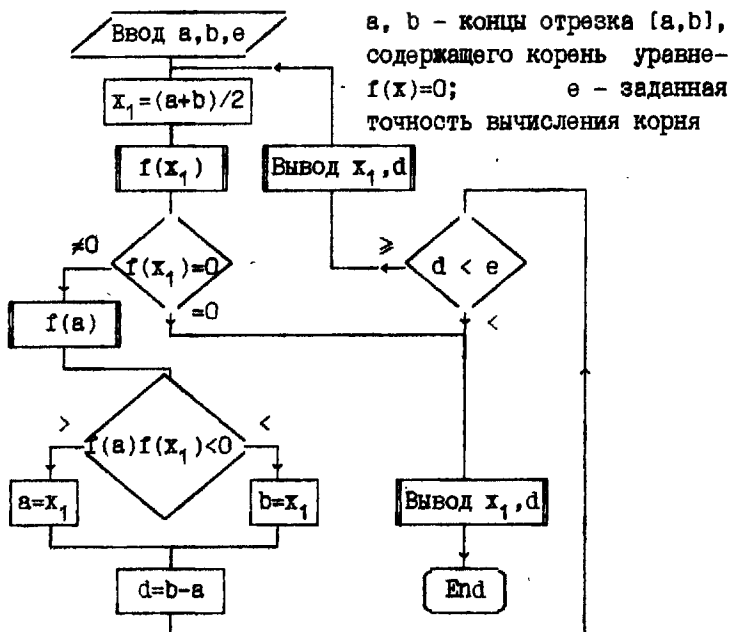
/* ----- */
    Выполнение последовательных итераций                */
----- */
do
{
    vectorofiteration();
    for (i = 1; i <= n; x[i] = x1[i] - x[i], i++);
    d = normlm(x);
    outputofvector(); pausa();
    changexbyx1();
} while (d*s >= e);
finish;;
}

```


ПРИЛОЖЕНИЕ К ГЛАВЕ 3

К §2

1°. Блок-схема решения уравнения $f(x)=0$ методом половинного деления



В приведенных программах рассматривается функция
 $f(x) = 4 - e^x - 2x^2$ (см. пример из §2).

2°. Программа решения уравнения $f(x)=0$ методом половинного деления на языке BASIC

```

1  REM *****
2  REM РЕШЕНИЕ УРАВНЕНИЯ f(x)=0 МЕТОДОМ ПОЛОВИННОГО ДЕЛЕНИЯ
3  REM *****
10 DEF fnf (x) = 4 - EXP(x) - 2 * x ^ 2
20 CLS
30 PRINT "Наберите через запятую три числа"
```

```

40 PRINT "координаты концов отрезка a, b , точность epsilon"
50 INPUT a, b, e: IF fnf(a) * fnf(b) > 0 THEN 150
60 x1 = (a + b) / 2: f1 = fnf(x1)
70 IF f1 <> 0 THEN 90
80 d = 0: GOSUB 160: GOTO 150
90 f = fnf(a)
100 IF f * f1 < 0 THEN b = x1: GOTO 120
110 a = x1
120 d = b - a
130 IF d > e THEN GOSUB 160: GOTO 60
140 GOSUB 160
150 END
151 REM -----
152 REM ПОДПРОГРАММА
153 REM Вывод результата вычислений на экран
160 CLS
170 PRINT "x = "; x1
180 PRINT "Погрешность результата"; d
190 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
200 IF INKEY$ = "" THEN 200
210 RETURN

```

**3⁰. Программа решения уравнения $f(x)=0$
методом половинного деления на языке PASCAL**

```

program SolutionOfEquationByDevision;
{ *****
РЕШЕНИЕ УРАВНЕНИЯ f(x)=0 МЕТОДОМ ПОЛОВИННОГО ДЕЛЕНИЯ
***** }
uses Crt;
label
    finish;
var
    a,b,d,e,x1,fa,f1:real;
    ch:char;
{ -----
ПОДПРОГРАММЫ
}
function f(x:real):real;

```

```

BEGIN
  f := 4 - EXP(x) - 2 * x * x
END;
{ ----- }
procedure OutputOfResult;
BEGIN
  ClrScr;
  WRITELN ('x =', x1);
  WRITELN ('Погрешность результата =', d);
  WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
  REPEAT ch := readkey UNTIL ch <> '';
  END;
{ ----- }
                                ОСНОВНАЯ ПРОГРАММА                                }
BEGIN
  ClrScr;
  WRITELN('          Введите по порядку три числа');
  WRITELN('координаты концов отрезка  a,  b ,  точность epsilon');
  READLN(a, b, e); IF f(a)*f(b) > 0 THEN GOTO finish;
  REPEAT
    x1 := (a + b) / 2; f1 := f(x1);
    IF f1 = 0 THEN
      BEGIN
        d := 0;
        OutputOfResult;
        GOTO finish;
      END;
    fa := f(a);
    IF fa * f1 < 0 THEN b := x1 ELSE a := x1;
    d := b-a;
    OutputOfResult;
  UNTIL d < e;
  finish:END.

```

4⁰. Программа решения уравнения $f(x)=0$
методом половинного деления на языке FORTRAN

```
$INCLUDE: 'EXEC.PI'
```

```
C *****
```

```

C  РЕШЕНИЕ УРАВНЕНИЯ  $f(x)=0$  МЕТОДОМ ПОЛОВИННОГО ДЕЛЕНИЯ
C  *****
INTEGER*2 system
f(x) = 4 - EXP(x) - 2 * x * x
i=system('cls'C)
WRITE (*,*) '    Наберите три числа: координаты'
WRITE (*,*) '    концов отрезка a, b, точность epsilon'
READ (*,*) a,b,e
IF (f(a)*f(b).GT.0) GOTO 1
WRITE (*,*)
    d = b - a
DO WHILE (d.GT.E)
    x1 = (a + b) / 2
    f1 = f(x1)
    IF (f1.EQ.0) THEN
        d = 0
        i=system('cls'C)
        CALL OutputOfResult(x1,d)
        GOTO 1
    END IF
    fa = f(a)
    IF (fa * f1.LT.0) THEN
        b = x1
    ELSE
        a = x1
    END IF
    d = b - a
    i=system('cls'C)
    CALL OutputOfResult(x1,d)
END DO
1 END

```

```

C  -----
C  ПОДПРОГРАММА
C  Вывод результата вычислений на экран
SUBROUTINE OutputOfResult(x1,d)
WRITE (*,'(A4,E12.6)') ' x =',x1
WRITE (*,'(A27,E12.6)') ' Погрешность результата =',d
PAUSE 'Нажмите клавишу ENTER для продолжения...'

```

RETURN
END

5⁰. Программа решения уравнения $f(x)=0$
методом половинного деления на языке C

```

/*****
РЕШЕНИЕ УРАВНЕНИЯ f(x)=0 МЕТОДОМ ПОЛОВИННОГО ДЕЛЕНИЯ
*****/
#include <stdio.h>
#include <conio.h>
#include <math.h>
float d,x1;
/* -----
                                ПОДПРОГРАММЫ                                */
float f(float x)
{ float y; y = 4 - exp(x) - 2 * x * x; return y; }
/* -----*/
void outputofresult(void)
{ clrscr();
  printf ("x = %f\n",x1);
  printf ("Погрешность приближения d = %f\n",d);
  printf ("\nНажмите любую клавишу для продолжения...");
  getch();
}
/* -----
                                ОСНОВНАЯ ПРОГРАММА                                */
void main ()
{ float a,b,e,fa,fi;
  char finish;
  clrscr();
  printf ("          Введите по порядку три числа n");
  printf ("координаты концов отрезка a, b, точность epsilon\n");
  scanf ("%f%f%f",&a, &b, &e);
  if (f(a)*f(b) > 0) goto finish;
do
{ x1 = (a + b) / 2; fi = f(x1);
  if (fi == 0)

```

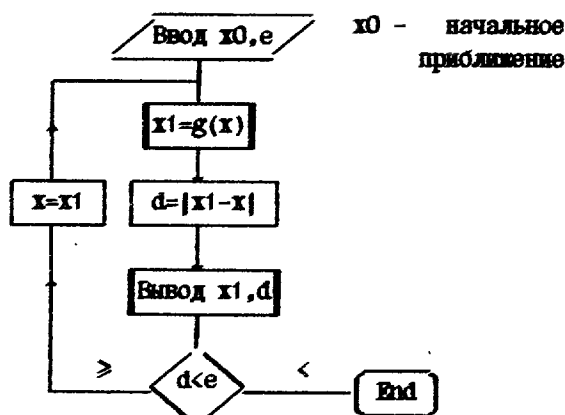
```

( d = 0;
  outputofresult();
  goto finish; );
fa = f(a);
if (fa * f1 < 0) b = x1; else a = x1;
d = b-a;
outputofresult();
)
while (d >= e);
finish;;
)

```

К §3

1⁰. Блок-схема решения уравнения $f(x)=0$
методом итераций



Примечание. Уравнение $f(x)=0$ должно быть предварительно преобразовано к эквивалентному уравнению $x = g(x)$.

В приведенных программах рассматривается функция $g(x) = \frac{1}{3} (2x - \ln x)$ (см. пример 2 §3).

Для достижения заданной точности результата ϵ вводится величина $e = \epsilon/10$.

2⁰. Программа решения уравнения $f(x) = 0$
методом итераций на языке BASIC

```

1  REM *****
2  REM    РЕШЕНИЕ УРАВНЕНИЯ f(x)=0 МЕТОДОМ ИТЕРАЦИИ
3  REM          f(x)=0 <==>  x = g(x)
4  REM *****
10 DEF fng (x) = (2 * x - LOG(x)) / 3
20 CLS
30 PRINT "Введите через запятую начальное значение x0 "
40 PRINT "          и точность e = еpsilon/10"
50 INPUT x, e
60 x1 = fng(x)
70 d = ABS(x1 - x)
80 GOSUB 110
90 IF d >= e THEN x = x1: GOTO 60
100 END
101 REM -----
102 REM          ПОДПРОГРАММА
110 CLS : PRINT "x ="; x1; " Погрешность приближения d ="; d
120 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
130 IF INKEY$ = "" THEN 130
140 RETURN

```

3⁰. Программа решения уравнения $f(x) = 0$
методом итераций на языке PASCAL

program SolutionOfEquationByIteration;

```

( *****
  РЕШЕНИЕ УРАВНЕНИЯ f(x)=0 МЕТОДОМ ИТЕРАЦИИ
    f(x)=0 <==>  x = g(x)
  ***** )

```

uses Crt;

var

d,e,x,x1:real;

ch:char;

```

( -----
                                ПОДПРОГРАММЫ
)

```

function g(x:real):real;

BEGIN

g := (2 * x - LN(x)) / 3

```

END;
procedure OutputOfResult;
  BEGIN
    ClrScr;
    WRITELN('x =', x1, ' Погрешность приближения d =', d);
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
  END;

```

```

{ -----
                                ОСНОВНАЯ ПРОГРАММА
}

```

```

BEGIN
  ClrScr;
  WRITELN('          Введите по порядку два числа');
  WRITELN('начальное значение x0 и точность e = epsilon/10');
  READLN(x,e);
  REPEAT
    x1:=g(x); d := ABS(x1 - x);
    OutputOfResult;
    x := x1;
  UNTIL d < e;
END.

```

4⁰. Программа решения уравнения $f(x) = 0$ методом итераций на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
  program SolutionOfEquationByIteration
C *****
C РЕШЕНИЕ УРАВНЕНИЯ  $f(x)=0$  МЕТОДОМ ИТЕРАЦИИ
C  $f(x)=0 \iff x = g(x)$ 
C *****
  INTEGER*2 system
  g(x) = (2 * x - LOG(x)) / 3
  i=system('cls'C)
  WRITE (*,*) '          Введите по порядку два числа'
  WRITE (*,*) 'начальное значение x0 и точность e=epsilon/10'
  READ (*,*) x,e
  WRITE (*,*)
    d = 1
  DO WHILE (d.GT.E)

```



```

      x1 = g(x)
      d = ABS(x1 - x)
      i=system('cls'C)
      CALL OutputOfResult(x1,d)
      x = x1
    END DO
  END

```

```

C -----
C                               ПОДПРОГРАММА
C   Вывод результата вычислений на экран
SUBROUTINE OutputOfResult(x1,d)
  WRITE (*,'(A4,E12.6)') ' x =',x1
  WRITE (*,'(A25,E12.6)') ' Погрешность приближения d =',d
  PAUSE 'Нажмите клавишу ENTER для продолжения...'
  RETURN
END

```

5⁰. Программа решения уравнения $f(x) = 0$ методом итераций на языке C

```

/* *****
РЕШЕНИЕ УРАВНЕНИЯ  $f(x)=0$  МЕТОДОМ ИТЕРАЦИИ
 $f(x)=0 \iff x = g(x)$ 
*****

```

```

#include <stdio.h>
#include <conio.h>
#include <math.h>
float d,x1;

```

```

/* -----
                               ПОДПРОГРАММЫ                               */
float g(float x)
{ float y; y = (2 * x - log(x)) / 3; return y; }
/* -----*/
void outputofresult(void)
{ clrscr();
  printf("x = %f\n",x1);
  printf("Погрешность приближения d = %f\n",d);
  printf("\nНажмите любую клавишу для продолжения...");
  getch();
}

```

```

/* -----
ОСНОВНАЯ ПРОГРАММА
*/

void main()
{ float e,x;
  clrscr();
  printf ("      Введите по порядку два числа n");
  printf ("начальное значение ");
  printf ("x0 и точность e=epsilon/10\n");
  scanf ("%f%f",&x,&e);
  do { x1=g(x); d = fabs(x1 - x);
      outputofresult();
      x = x1; }
  while (d >= e);
}

```

К §4

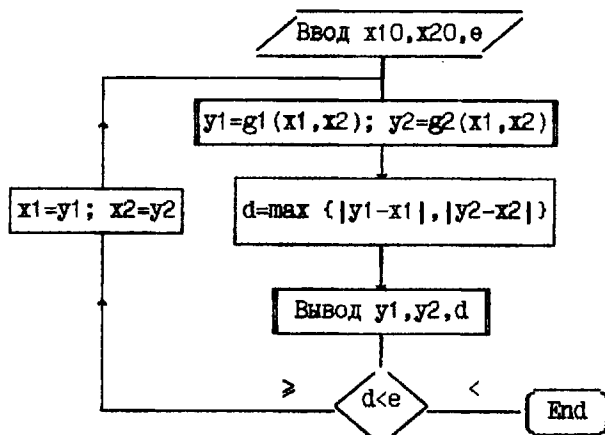
1⁰. Блок-схема численного решения
системы нелинейных уравнений методом итераций

Система уравнений

$$\begin{cases} f_1(x_1, x_2) = 0, \\ f_2(x_1, x_2) = 0 \end{cases}$$

должна быть предварительно преобразована к эквивалентной
системе уравнений

$$\begin{cases} x_1 = g_1(x_1, x_2), \\ x_2 = g_2(x_1, x_2). \end{cases}$$



В приведенных программах рассматриваются функции

$$g_1(x_1, x_2) = x_1 - \frac{1}{2} (x_2(x_1 - 1) - 1) - \frac{1}{12} (x_1^2 - x_2^2 - 1),$$

$$g_2(x_1, x_2) = x_2 - \frac{1}{2} (x_2(x_1 - 1) - 1) + \frac{1}{4} (x_1^2 - x_2^2 - 1)$$

(см. пример из §4).

Окончание процесса итераций определяется величиной $\epsilon = \epsilon/10$, где ϵ — точность результата.

**2⁰. Программа численного решения
системы нелинейных уравнений методом итераций
на языке BASIC**

```

1  REM *****
2  REM РЕШЕНИЕ СИСТЕМЫ НЕЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ ИТЕРАЦИИ
3  REM      f1(x1,x2) = 0                      x1 = g1(x1,x2)
4  REM      < === >
5  REM      f2(x1,x2) = 0                      x1 = g2(x1,x2)
6  REM *****
10 DEF fnf1 (x1, x2) = x2 * (x1 - 1) - 1
20 DEF fnf2 (x1, x2) = x1 * x1 - x2 * x2 - 1
30 DEF fng1 (x1, x2) = x1 - fnf1(x1, x2) / 2 - fnf2(x1, x2) / 12
40 DEF fng2 (x1, x2) = x2 - fnf1(x1, x2) / 2 + fnf2(x1, x2) / 4
50  CLS
60  PRINT "Введите через запятую начальные значения "
70  PRINT "  x10,  x20  и  точность e = epsilon/10"
80  INPUT x1, x2, e
90  y1 = fng1(x1, x2): y2 = fng2(x1, x2)
100 d = ABS(y1 - x1)
110 IF ABS(y2 - x2) > d THEN d = ABS(y2 - x2)
120 GOSUB 150
130 IF d >= e THEN x1 = y1: x2 = y2: GOTO 90
140 END
141 REM -----
142 REM                      ПОДПРОГРАММА
143 REM -----

```

```

150 CLS
160 PRINT "x1 ="; y1; " Погрешность приближения ="; d
170 PRINT "x2 ="; y2
180 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
190 IF INKEY$ = "" THEN 190
200 RETURN

```

3⁰. Программа численного решения
системы нелинейных уравнений методом итераций
на языке PASCAL

```

program SolutionOfSystemByIterat;
( *****
РЕШЕНИЕ СИСТЕМЫ НЕЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ ИТЕРАЦИИ
      f1(x1,x2) = 0                x1 = g1(x1,x2)
                                < === >
      f2(x1,x2) = 0                x2 = g2(x1,x2)
***** )
uses Crt;
var
  d,e,x1,x2,y1,y2:real;
  ch:char;
{ -----
                                ПОДПРОГРАММЫ                                }
procedure fun_y1_y2 ;
var
  f1,f2:real;
BEGIN
  f1 := x2 * (x1 - 1) - 1;
  f2 := x1 * x1 - x2 * x2 - 1;
  y1 := x1 - f1 / 2 - f2 / 12;
  y2 := x2 - f1 / 2 + f2 / 4;
END;
procedure OutputOfResult;
BEGIN
  ClrScr;
  WRITELN('x1 =', y1, ' Погрешность приближения =', d);
  WRITELN('x2 =', y2);

```

```
WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
REPEAT ch := readkey UNTIL ch <> '';
END;
```

ОСНОВНАЯ ПРОГРАММА

```
BEGIN
  ClrScr;
  WRITELN('      Введите по порядку три числа');
  WRITELN('начальные значения x10, x20 и точность e=epsilon/10');
  READLN(x1,x2,e);
  REPEAT
    fun_y1_y2;
    d := ABS(y1 - x1);
    IF ABS(y2-x2) > d THEN d := ABS(y2-x2);
    OutputOfResult;
    x1 := y1; x2 := y2;
  UNTIL d < e;
END.
```

4⁰. Программа численного решения системы нелинейных уравнений методом итераций на языке FORTRAN

```
$INCLUDE: 'EXEC.F1'
  program SolutionOfSystemByIteration
C *****
C РЕШЕНИЕ СИСТЕМЫ НЕЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ ИТЕРАЦИИ
C      f1(x1,x2) = 0      x1 = g1(x1,x2)
C      < == >
C      f2(x1,x2) = 0      x1 = g2(x1,x2)
C *****
  INTEGER*2 system
  f1(x1, x2) = x2 * (x1 - 1) - 1
  f2(x1, x2) = x1 * x1 - x2 * x2 - 1
  g1(x1, x2) = x1 - f1(x1, x2) / 2 - f2(x1, x2) / 12
  g2(x1, x2) = x2 - f1(x1, x2) / 2 + f2(x1, x2) / 4
  1=system('cls'C)
  WRITE (*,*) '      Введите начальные значения'
```

```

WRITE (*,*) ' x10, x20 и точность e = epsilon/10'
READ (*,*) x1,x2,e
WRITE (*,*)
  d = 1
DO WHILE (d.GT.E)
  y1 = g1(x1,x2)
  y2 = g2(x1,x2)
  d = ABS(y1 - x1)
  IF (ABS(y2 - x2).GT.d) d = ABS(y2 - x2)
  i=system('cls'C)
  CALL OutputOfResult(y1,y2,d)
  x1 = y1
  x2 = y2
END DO
END

```

C

C

C

ПОДПРОГРАММА

```

Вывод результата вычислений на экран
SUBROUTINE OutputOfResult(y1,y2,d)
WRITE (*, '(A5,E12.6)') ' x1 =',y1
WRITE (*, '(A5,E12.6)') ' x2 =',y2
WRITE (*, '(A25,E12.6)') ' Погрешность приближения d =',d
PAUSE 'Нажмите клавишу ENTER для продолжения...'
RETURN
END

```

5⁰. Программа численного решения
системы нелинейных уравнений методом итераций
на языке C

```

/* *****
РЕШЕНИЕ СИСТЕМЫ НЕЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ ИТЕРАЦИИ
f1(x1,x2) = 0          x1 = g1(x1,x2)
                      < == >
f2(x1,x2) = 0          x2 = g2(x1,x2)
*****/
#include <stdio.h>
#include <conio.h>

```

```
#include <math.h>
float d,x1,x2,y1,y2;
```

```
/* -----
                                ПОДПРОГРАММЫ                                */
```

```
void fun_y1_y2(void)
{ float f1,f2;
  f1 = x2 * (x1 - 1) - 1;
  f2 = x1 * x1 - x2 * x2 - 1;
  y1 = x1 - f1 / 2 - f2 / 12;
  y2 = x2 - f1 / 2 + f2 / 4;
}
```

```
/* -----*/
void outputofresult(void)
{ clrscr();
  printf ("x1 = %f Погрешность приближения d = %f",y1,d);
  printf ("\nx2 = %f",y2);
  printf ("\n\nНажмите любую клавишу для продолжения...");
  getch();
}
```

```
/* -----
                                ОСНОВНАЯ ПРОГРАММА                                */
```

```
void main()
{ float e;
  clrscr();
  printf ("          Введите по порядку три чиола n");
  printf("начальные значения x10, x20 и ");
  printf ("точность e=epsilon/10\n");
  scanf ("%f%f%f",&x1,&x2,&e);
  do { fun_y1_y2();
      d = fabs(y1 - x1);
      if (fabs(y2-x2) > d) d = fabs(y2-x2);
      outputofresult();
      x1 = y1; x2 = y2;}
  while (d >= e);
}
```

ПРИЛОЖЕНИЕ К ГЛАВЕ 4

К §2

1⁰. Блок-схема построения тригонометрического многочлена, аппроксимирующего заданную функцию

Тригонометрический многочлен степени n наилучшего среднеквадратического приближения функции

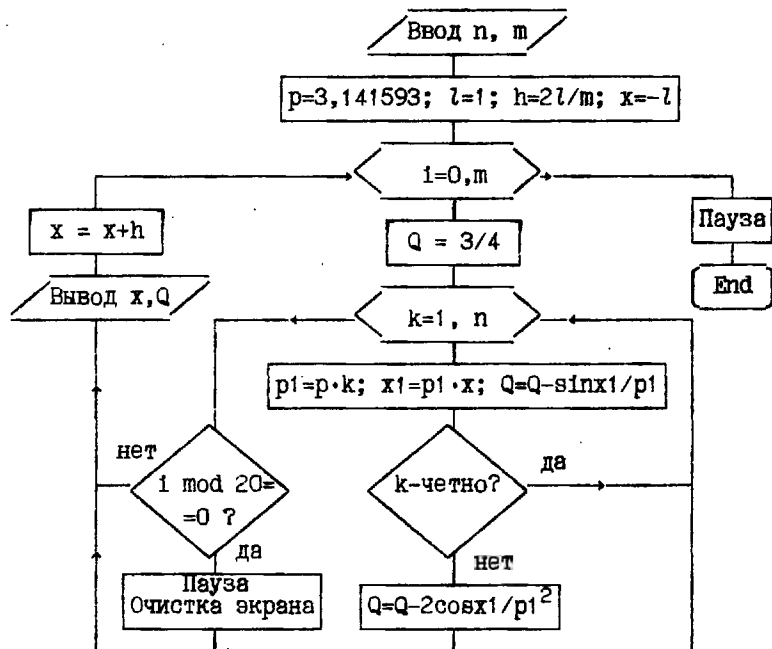
$$f(x) = \begin{cases} 1, & -1 \leq x < 0; \\ x, & 0 < x \leq 1 \end{cases}$$

имеет вид

$$Q_n(x) = \frac{3}{4} - \sum_{k=1}^n (1 - (-1)^k) \frac{1}{\pi^2} \frac{\cos \pi k x}{k^2} + \frac{1}{\pi} \frac{\sin \pi k x}{k}$$

(см. пример 1 из §2).

Пусть требуется вычислить значения многочлена $Q_n(x)$ на множестве равноотстоящих точек с шагом $h = 2l/m$ на отрезке $[-l, l]$ (в данном примере $l=1$). На экран выводить не более 20 значений. Блок-схема этой задачи такова:



2°. Программа построения тригонометрического многочлена, аппроксимирующего заданную функцию, на языке BASIC

```

1 REM ***** РАСЧЕТ *****
2 REM ТАБЛИЦЫ ЗНАЧЕНИЙ ТРИГОНОМЕТРИЧЕСКОГО МНОГОЧЛЕНА
3 REM НАИЛУЧШЕГО СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
4 REM *****
10 DEFINT I, K, M-N: CLS
20 PRINT "      Введите через запятую два числа: "
30 PRINT "n-порядок многочлена, m-число разбиений отрезка [-1,1]"
40 INPUT n, m: CLS
50 p = 3.141593: l = 1: h = 2 * l / m: x = -1
60 FOR i = 0 TO m
70   Q = .75
80   FOR k = 1 TO n
90     p1 = p * k: x1 = p1 * x: Q = Q - SIN(x1) / p1
100    IF k MOD 2 = 1 THEN Q = Q - 2 * COS(x1) / p1 ^ 2
110  NEXT k
120  IF i <> 0 AND i MOD 20 = 0 THEN GOSUB 170: CLS
130  PRINT USING "x = ##.#### Q## = ##.####": x, n; Q
140  x = x + h
150 NEXT i
160 GOSUB 170: END
170 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
180 IF INKEY$ = "" THEN 180
190 RETURN

```

3°. Программа для построения тригонометрического многочлена, аппроксимирующего заданную функцию, на языке PASCAL

```

PROGRAM FOURIEpolinom;
(
  ***** РАСЧЕТ *****
  ТАБЛИЦЫ ЗНАЧЕНИЙ ТРИГОНОМЕТРИЧЕСКОГО МНОГОЧЛЕНА
  НАИЛУЧШЕГО СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
  *****
)
uses Crt;
var

```

```

i,k,m,n:integer;
h,l,Q,p1,x,x1:real;
ch:char;
{ -----
                                ПОДПРОГРАММА                                }
procedure PAUSA;
BEGIN
  WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
  REPEAT ch := readkey UNTIL ch <> '';
END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                }
BEGIN
  ClrScr;
  WRITELN (' Введите  два  числа: n-порядок многочлена,');
  WRITELN ('          m-число разбиений отрезка (-l,l)');
  READ (n, m);
  ClrScr;
  l := 1; h := 2 * l / m; x := -1;
  FOR i := 0 TO m DO
    BEGIN
      Q := 0.75;
      FOR k := 1 TO n DO
        BEGIN
          p1 := p1 * k; x1 := p1 * x; Q := Q - SIN(x1) / p1;
          IF k MOD 2 = 1 THEN Q := Q - 2 * COS(x1) / (p1*p1);
        END;
      IF (i<>0) and (i MOD 20 = 0) THEN
        BEGIN
          PAUSA; ClrScr;
        END;
      WRITELN('x =',x:8:5,'  Q',n:2,' =',Q:8:5);
      x := x + h;
    END;
  PAUSA;
END.

```

4⁰. Программа для построения тригонометрического многочлена, аппроксимирующего заданную функцию, на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
PROGRAM FOURIEpolinom
C ***** РАСЧЕТ *****
C ТАБЛИЦЫ ЗНАЧЕНИЙ ТРИГОНОМЕТРИЧЕСКОГО МНОГОЧЛЕНА
C НАИЛУЧШЕГО СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
C *****
REAL 1
INTEGER*2 system
i=system('cls'C)
WRITE(*,*) ' Введите два числа: n-порядок многочлена,'
WRITE(*,*) '          m-число разбиений отрезка [-1,1]'
READ (*,*) n, m
i=system('cls'C)
  p = 3.141593
  l = 1
  h = 2 * l / m
  x = -1
DO 1 = 0,m
  Q = .75
  DO k = 1,n
    p1 = p * k
    x1 = p1 * x
    Q = Q - SIN(x1) / p1
    IF (MOD(k,2).EQ.1) Q = Q - 2 * COS(x1) /p1**2
  END DO
  IF (1.NE.0 .AND. MOD(1,20).EQ.0) THEN
    PAUSE ' Нажмите клавишу ENTER для продолжения...'
    k=system('cls'C)
  END IF
  WRITE (*, '(A3,E12.6,A2,I2,A1,E12.6)') ' x=',x,' Q ',n,'=',Q
  x = x + h
END DO
  PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

**5⁰. Программа для построения тригонометрического многочлена,
аппроксимирующего заданную функцию, на языке C**

```

/* ***** РАСЧЕТ *****
   ТАБЛИЦЫ ЗНАЧЕНИЙ ТРИГОНОМЕТРИЧЕСКОГО МНОГОЧЛЕНА
   НАИЛУЧШЕГО СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
   *****/
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define Pi 4*atan(1)
/*          ПОДПРОГРАММА          */
void pausa()
{ printf("\nНажмите любую клавишу для продолжения...");
  getch();
}
/* -----
          ОСНОВНАЯ ПРОГРАММА          */
void main()
{ int i,k,m,n;
  float h,l,Q,p1,x,x1 ;
  clrscr();
  printf (" Введите два числа: n-порядок многочлена,\n");
  printf ("      m-число разбиений отрезка [-1,1]\n");
  scanf ("%i%i",&n, &m);
  for (clrscr(),l=1,h=2*l/m,x=-1, i = 0; i <= m; i++)
  { for (Q = 0.75,k = 1; k <= n; k++)
    { p1 = Pi * k; x1 = p1 * x; Q = Q - sin(x1) / p1;
      if (k % 2 == 1) Q = Q - 2 * cos(x1) / (p1*p1);
    };
    if (i != 0 && i % 20 == 0)
    { pausa(); clrscr(); };
    printf ("x = %f   Q%i = %f\n",x,n,Q);
    x += h;
  };
  pausa();
}

```

К §4

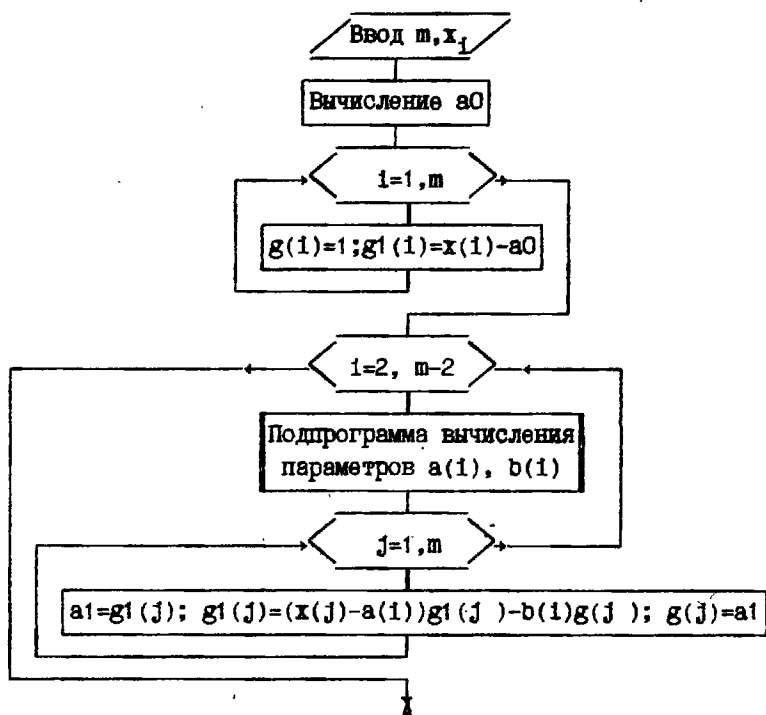
1⁰. Блок-схема вычисления
ортогональных многочленов Чебышева
на заданном множестве точек

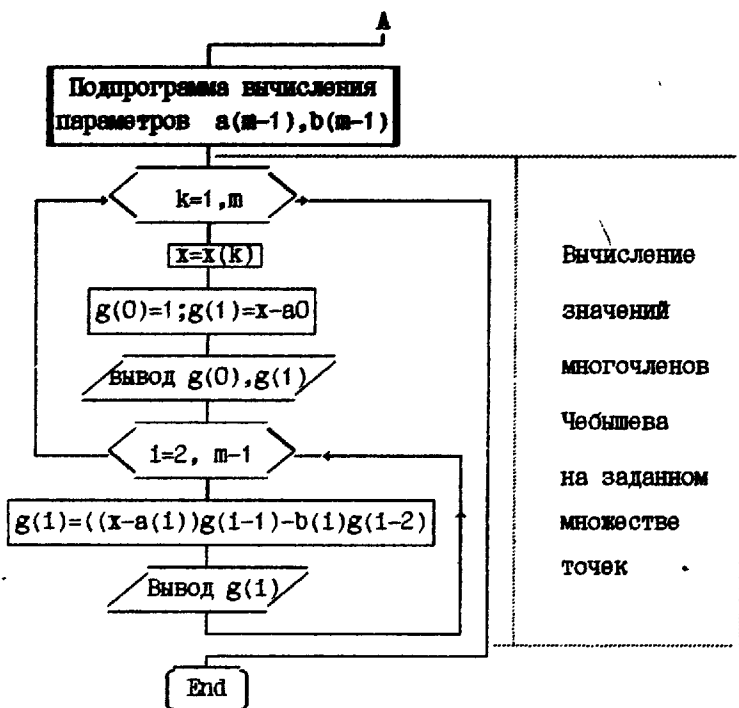
Пусть задано множество точек (x_i) ($i=1, 2, \dots, m$). Рекуррентные соотношения, определяющие ортогональные многочлены Чебышева $g_k(x)$ ($k=0, 1, \dots, m-1$), имеют вид

$$g_0=1; \quad g_1(x)=x-a; \quad g_k(x)=(x-a_k)g_{k-1}(x)-b_k g_{k-2}(x) \quad (k=2, 3, \dots, m-1).$$

Значения коэффициентов $a_0=a$, a_k , b_k находятся по формулам

$$a = \frac{1}{m} \sum_{i=1}^m x_i, \quad a_k = \frac{\sum_{i=1}^m x_i g_{k-1}^2(x_i)}{\sum_{i=1}^m g_{k-1}^2(x_i)}, \quad b_k = \frac{\sum_{i=1}^m x_i g_{k-2}(x_i) g_{k-1}(x_i)}{\sum_{i=1}^m g_{k-2}^2(x_i)}.$$





2⁰. Программа вычисления ортогональных
многочленов Чебышева на языке BASIC

```

1 REM *****
2 REM ПОСТРОЕНИЕ ОРТОГОНАЛЬНЫХ МНОГОЧЛЕНОВ ЧЕБЫШЕВА
3 REM НА МНОЖЕСТВЕ m < 16 ТОЧЕК
4 REM *****
10 DEFINT I-J, M: DIM x(15), g(15), g1(15), a(14), b(14)
20 CLS
30 INPUT "Введите m - число точек множества (m < 16) "; m
40 PRINT "Введите последовательно значения x1; i = 1, ..., m"
50 a0 = 0
60 FOR i = 1 TO m
70 INPUT x(i): a0 = a0 + x(i)
80 NEXT i

```

```

90  a0 = a0 / m
100  FOR i = 1 TO m
110    g(1) = 1: g1(1) = x(1) - a0
120  NEXT i
130  FOR i = 2 TO m - 2
140    GOSUB 320
150    FOR j = 1 TO m
160      a1 = g1(j)
170      g1(j) = (x(j) - a(1)) * g1(j) - b(1) * g(j)
180      g(j) = a1
190    NEXT j
200  NEXT i: GOSUB 320
210  FOR k% = 1 TO m
220    x = x(k%)
230    g(0) = 1: g(1) = x - a0
240    PRINT USING "x = ##.## "; x;
250  PRINT USING "  g0 = ##.###  g1 = ##.###"; g(0); g(1);
260    FOR i = 2 TO m - 1
270      g(i) = (x - a(1)) * g(i - 1) - b(1) * g(i - 2)
280      PRINT USING "  g# = ##.###"; i; g(i);
290    NEXT i: PRINT
300  NEXT k%
310  GOSUB 390: END
311  REM -----
312  REM              ПОДПРОГРАММЫ
313  REM  Вычисление коэффициентов a(1), b(1) рекуррентных
314  REM  соотношений для многочленов Чебышева
315  REM -----
320  k0 = 0: n1 = 0: b0 = 0: n0 = 0
330  FOR j = 1 TO m
340    k0 = k0 + x(j) * g1(j) ^ 2: n1 = n1 + g1(j) ^ 2
350    b0 = b0 + x(j) * g(j) * g1(j): n0 = n0 + g(j) ^ 2
360  NEXT j
370  a(1) = k0 / n1: b(1) = b0 / n0
380  RETURN
381  REM -----
382  REM              Пауза
390  PRINT : PRINT "Нажмите любую клавишу для продолжения"

```

```

400 IF INKEY$ = "" THEN 400
410 RETURN

```

3⁰. Программа вычисления ортогональных многочленов Чебышева на языке PASCAL

```

program PolinomOfTchebishev;
(
  *****
  ПОСТРОЕНИЕ ОРТОГОНАЛЬНЫХ МНОГОЧЛЕНОВ ЧЕБЫШЕВА
  НА МНОЖЕСТВЕ m < 16 ТОЧЕК
  ***** )
uses Crt;
const
  l=15;
type
  data=array [1..l] of real;
  functions=array [0..l] of real;
  parameters=array [2..l-1] of real;
var
  i,j,k,m:integer;
  a0,a1,x1:real;
  a,b:parameters;
  x:data;
  g,g1:functions;
  ch:char;
(
  -----
  - ПОДПРОГРАММЫ
  ----- )

procedure PAUSA;
BEGIN
  WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
  REPEAT ch := readkey UNTIL ch <> '';
END;
(
  -----
  Вычисление параметров рекуррентной формулы a[k], b[k]
  )
procedure CalculOfParameters(i:integer);
var
  k0,b0,n0,n1:real;

```



```

FOR k := 1 TO m DO
  BEGIN
    x1 := x[k];
    g[0] := 1; g[1] := x1 - a0;
    WRITE ('x = ',x1:5:2,'   g0 = ',g[0]:6:3,'   g1 = ',g[1]:6:3);
    FOR i := 2 TO m - 1 DO
      BEGIN
        g[i] := (x1 - a[i]) * g[i - 1] - b[i] * g[i - 2];
        WRITE ('   g',i:1,' = ',g[i]:6:3);
      END;
      Writeln;
    END;
    PAUSA;
  END.

```

4⁰. Программа вычисления ортогональных многочленов Чебышева на языке FORTRAN

```

$INCLUDE: 'EXEC.PI'
program PolinomOfTchebishev
C *****
C ПОСТРОЕНИЕ ОРТОГОНАЛЬНЫХ МНОГОЧЛЕНОВ ЧЕБЫШЕВА
C НА МНОЖЕСТВЕ m < 16 ТОЧЕК
C *****
INTEGER*2 system
COMMON m,x(15),g(0:15),g1(0:15),a(2:14),b(2:14)
i=system('cls'C)
WRITE (*,*) 'Введите m - число точек множества (m < 16)'
READ (*,*) m
WRITE (*,*) 'Введите значения x1; i=1,...,m'
  a0 = 0
  DO i=1,m
    READ (*,*) x(i)
    a0 = a0 + x(i)
  END DO
  a0 = a0 / m
  DO i = 1,m
    g(i) = 1
    g1(i) = x(i) - a0
  END DO

```

```

DO i = 2, m - 2
  CALL CalculOfParameters (i)
  DO j = 1,m
    a1 = g1(j)
    g1(j) = (x(j) - a(1)) * g1(j) - b(1) * g(j)
    g(j) = a1
  END DO
END DO
CALL CalculOfParameters (m - 1)
DO k = 1,m
  x1 = x(k)
  g(0) = 1
  g(1) = x1 - a0
WRITE (*, '(A4,F5.2\)' ) ' x =',x1
WRITE (*, '(A5,F6.3,A5,F6.3\)' ) ' g0=',g(0), ' g1=',g(1)
  DO i = 2, m - 1
    g(i) = (x1 - a(1)) * g(i - 1) - b(1) * g(i - 2)
    WRITE (*, '(A4,I1,A2,F6.3\)' ) ' g',i,'=' ,g(i)
  END DO
  WRITE (*,*)
END DO
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

C
C
C
C

ПОДПРОГРАММА

Вычисление параметров рекуррентной формулы $a(k)$, $b(k)$

```

SUBROUTINE CalculOfParameters (i)
REAL k0,n0,n1
COMMON m,x(15),g(0:15),g1(0:15),a(2:14),b(2:14)
k0 = 0
n1 = 0
b0 = 0
n0 = 0
DO j = 1,m
  k0 = k0 + x(j) * g1(j) * g1(j)
  n1 = n1 + g1(j) * g1(j)
  b0 = b0 + x(j) * g(j) * g1(j)

```

```

    n0 = n0 + g(j) * g(j)
END DO
a(1) = k0 / n1
b(1) = b0 / n0
RETURN
END

```

5⁰. Программа вычисления ортогональных многочленов Чебышева на языке C

```

/* *****
ПОСТРОЕНИЕ ОРТОГОНАЛЬНЫХ МНОГОЧЛЕНОВ ЧЕБЫШЕВА
НА МНОЖЕСТВЕ m < 16 ТОЧЕК
***** */

#include <stdio.h>
#include <conio.h>
#define L 15
int i,m;
float a[L],b[L],x[L+1],g1[L+1],g[L+1];
/* -----
                                ПОДПРОГРАММА
Вычисление параметров рекуррентной формулы a[k], b[k] */
void calculoofparameters(i)
int i;
{ int j;
  float k0,b0,n0,n1;
  for (k0=n1=b0=n0=0,j=1; j <= m; j++)
  { k0 += x[j] * g1[j] * g1[j]; n1 += g1[j] * g1[j];
    b0 += x[j] * g[j] * g1[j]; n0 += g[j] * g[j];
  };
  a[i] = k0 / n1; b[i] = b0 / n0;
}
/* -----
                                ОСНОВНАЯ                                ПРОГРАММА                                */

void main()
{ int j,k;
  float a0,a1,x1;
  clrscr();
  printf ("Введите m - число точек множества (m < 16)\n");

```

```

scanf ("%i",&m);
printf ("Введите последовательно значения xi i=1,...,m\n");
for (a0 = 0, i = 1; i <= m; i++)
{ scanf ("%f",&x[i]); a0 += x[i]; }
for (a0 /= m, i = 1; i <= m; i++)
{ g[i] = 1; g[i] = x[i] - a0; }
for (i = 2; i <= m - 2; i++)
{ calculofparameters(i);
  for (j = 1; j <= m; j++)
  { a1 = g[j];
    g[j] = (x[j] - a[i]) * g[j] - b[i] * g[j];
    g[j] = a1;
  };
  calculofparameters(m-1);
}
for (k = 1; k <= m; k++)
{ x1 = x[k]; g[0] = 1; g[1] = x1 - a0;
printf ("x = %5.2f g0 = %6.3f g1 = %6.3f", x1, g[0], g[1]);
  for (i = 2; i <= m - 1; i++)
  { g[i] = (x1 - a[i]) * g[i - 1] - b[i] * g[i - 2];
    printf (" g%i = %6.3f", i, g[i]);
  }; printf ("\n");
};
printf("\nНажмите любую клавишу для продолжения...");
getch();
}

```

1¹. Вычисление многочленов наилучшего среднеквадратичного приближения функций ортогональными многочленами

Пусть заданы функция в виде множества пар чисел $\{(x_i, y_i)\}$ ($i=1, 2, \dots, m$) и степень n ($0 \leq n \leq m-1$) многочлена $Q_n(x)$ наилучшего среднеквадратичного приближения. Рекуррентные соотношения, определяющие ортогональные многочлены Чебышева и многочлен $Q_n(x)$, имеют вид

$$g_0=1; \quad g_1(x)=x-a; \quad g_k(x)=(x-a_k)g_{k-1}(x)-b_k g_{k-2}(x) \quad (k=2, 3, \dots, n);$$

$$Q_0(x) = c_0, \quad Q_k(x) = Q_{k-1}(x) + C_k g_k(x) \quad (k=1, 2, \dots, n).$$

Значения коэффициентов a , a_k , b_k , C_k находятся по формулам

$$a = \frac{1}{n} \sum_{i=1}^n x_i, \quad a_k = \frac{\sum_{i=1}^n x_i g_{k-1}^2(x_i)}{\sum_{i=1}^n g_{k-1}^2(x_i)}, \quad b_k = \frac{\sum_{i=1}^n x_i g_{k-2}(x_i) g_{k-1}(x_i)}{\sum_{i=1}^n g_{k-2}^2(x_i)} \\ (k = 2, 3, \dots, n);$$

$$c_k = \frac{(f, g_k)}{\|g_k\|^2} = \frac{\sum_{i=1}^n y_i g_k(x_i)}{\sum_{i=1}^n g_k^2(x_i)} \quad (k=0, 1, 2, \dots, n).$$

Погрешность приближения оценивается величиной

$$\bar{d} = \sqrt{\frac{\sigma^2(f, Q_n)}{N^2}},$$

где

$$\sigma^2(f, Q_n) = \sum_{i=1}^n (y_i - Q_n(x_i))^2 = N^2 - \sum_{k=0}^n c_k^2 \|g_k\|^2, \quad N^2 = \sum_{i=1}^n y_i^2.$$

Блок-схема и программа вычисления многочлена наилучшего средне-квадратичного приближения представляют собой пополнение блок-схемы и программы вычисления значений ортогональных многочленов Чебышева. Программа позволяет вычислить значение многочлена $Q_n(x)$ в точках x_k ($k = 1, 2, \dots, n$).

**2¹. Программа вычисления
многочленов наилучшего среднеквадратичного
приближения функций ортогональными многочленами
на языке BASIC**

```

1 REM ***** РАСЧЕТ *****
2 REM ТАБЛИЦЫ ЗНАЧЕНИЙ СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
3 REM ФУНКЦИИ ОРТОГОНАЛЬНЫМИ МНОГОЧЛЕНАМИ ЧЕБЫШЕВА
4 REM *****
10 DIM x(15), y(15), g(15), g1(15), a(14), b(14), c(14)
20 DEFINT I-K, M-N: CLS
30 INPUT "Введите m - число значений функции (m < 16) "; m
40 INPUT "Введите n-порядок аппроксимирующего многочлена"; n
50 PRINT "    Вводите через запятую пары значений "
51 PRINT "        x1,y1; i=1,...,m"
60    a0 = 0: y1 = 0: y2 = 0

```

```

70  FOR i = 1 TO m
80    INPUT x(i), y(i)
90    a0 = a0 + x(i): y1 = y1 + y(i): y2 = y2 + y(i) ^ 2
100  NEXT
110  a0 = a0 / m: c(0) = y1 / m
120  IF n = 0 THEN 250
130  FOR i = 1 TO m
140    g(i) = 1: g1(i) = x(i) - a0
150  NEXT i
160  IF n > m - 1 THEN n = m - 1
170  FOR i = 2 TO n
180    GOSUB 380: GOSUB 440
190    FOR j = 1 TO m
200      a1 = g1(j)
210      g1(j) = (x(j) - a(1)) * g1(j) - b(1) * g(j)
220      g(j) = a1
230    NEXT j
240  NEXT i: GOSUB 380
250  d = 0
260  FOR k = 1 TO m
270    x = x(k)
280    g(0) = 1: g(1) = x - a0
290    g1(0) = c(0): g1(1) = g1(0) + c(1) * g(1)
300    FOR i = 2 TO n
310      g(i) = (x - a(1)) * g(i - 1) - b(1) * g(i - 2)
320      g1(i) = g1(i - 1) + c(1) * g(i)
330    NEXT i: d = d + (y(k) - g1(n)) ^ 2
340    PRINT USING "x = ##.##   Q# = ##.### "; x; n; g1(n)
350  NEXT k: d = SQR(d / y2)
360  PRINT USING "Погрешность приближения d = #.#####"; d
370  GOSUB 510: END
380  n1! = 0: c1 = 0
390  FOR j = 1 TO m
400    n1! = n1! + g1(j) ^ 2: c1 = c1 + y(j) * g1(j)
410  NEXT j
420  c(1 - 1) = c1 / n1!
430  RETURN
440  k0! = 0: b0 = 0: n0! = 0: c0 = 0

```

```

450  FOR j = 1 TO m
460      k0! = k0! + x(j) * g1(j) ^ 2:
470      b0 = b0 + x(j) * g(j) * g1(j): n0! = n0! + g(j) ^ 2
480  NEXT j
490      a(1) = k0! / n1!: b(1) = b0 / n0!
500  RETURN
510  PRINT : PRINT "Нажмите любую клавишу для продолжения"
520  IF INKEY$ = "" THEN 520
530  RETURN

```

**3¹. Программа вычисления
 многочленов наилучшего среднеквадратичного
 приближения функций ортогональными многочленами
 на языке PASCAL**

```

program ApproximationByPolinomsOfTchebishev;
{ ***** РАСЧЕТ *****
  ТАБЛИЦЫ ЗНАЧЕНИЙ СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
  ФУНКЦИИ ОРТОГОНАЛЬНЫМИ МНОГОЧЛЕНАМИ ЧЕБЫШЕВА
  ***** }
uses Crt;
const
    l=15;
type
    data=array [1..l] of real;
    functions=array [0..l] of real;
    coef=array [0..l-1] of real;
    parameters=array [2..l-1] of real;
var
    i,j,k,m,n:integer;
    a0,a1,d,n1,x1,y1,y2:real;
    a,b:parameters;
    x,y:data;
    g,g1:functions;
    c:coef;
    ch:char;
{ -----
  ПОДПРОГРАММЫ
  ----- }
procedure PAUSA;

```


BEGIN

WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');

REPEAT ch := readkey UNTIL ch <> '';

END;

{ -----
 Вычисление коэффициентов Фурье c[k] }

procedure CalculOfCoeffOfFourie(i:integer);

var

 c1:real;

BEGIN

 n1 := 0; c1 := 0;

 FOR j := 1 TO m DO

 BEGIN

 c1 := c1 + y[j] * g1[j]; n1 := n1 + g1[j] * g1[j];

 END;

 c1 - 11 := c1 / n1;

END;

{ -----
 Вычисление параметров рекуррентной формулы a[k], b[k] }

procedure CalculOfParameters(i:integer);

var

 k0,b0,n0:real;

BEGIN

 k0 := 0; b0 := 0; n0 := 0;

 FOR j := 1 TO m DO

 BEGIN

 k0 := k0 + x[j] * g1[j] * g1[j];

 b0 := b0 + x[j] * g[j] * g1[j];

 n0 := n0 + g[j] * g[j];

 END;

 a11 := k0 / n1; b11 := b0 / n0;

END;

{ -----
 ОСНОВНАЯ ПРОГРАММА }

BEGIN

 ClrScr;

 WRITELN ('Введите m - число значений функции (m < 16)');

 READ (m);

```

WRITELN ('Введите n - порядок аппроксимирующего многочлена');
READ (n);
WRITELN ('Вводите пары значений x1,y1; i=1,...,m');
  a0 := 0; y1 := 0; y2 := 0;
FOR i := 1 TO m DO
  BEGIN
    READ (x[i],y[i]); WRITELN;
    a0 := a0 + x[i]; y1 := y1 + y[i];
    y2 := y2 + y[i] * y[i];
  END;
  a0 := a0 / m; c[0] := y1 / m;
FOR i := 1 TO m DO
  BEGIN
    g[i] := 1; g[i] := x[i] - a0;
  END;
  IF n <> 0 THEN
  BEGIN
    IF n > m - 1 THEN n := m - 1;
    FOR i := 2 TO n DO
      BEGIN
        CalculOfCoeffOfFourie(1); CalculOfParameters(1);
        FOR j := 1 TO m DO
          BEGIN
            a1 := g[j];
            g[j] := (x[j] - a[1]) * g[j] - b[1] * g[j];
            g[j] := a1;
          END;
        END; CalculOfCoeffOfFourie(n+1);
      END;
    d:=0;
    FOR k := 1 TO m DO
      BEGIN
        x1 := x[k];
        g[0] := 1; g[1] := x1 - a0;
        g[0] := c[0]; g[1] := g[0] + c[1] * g[1];
        FOR i := 2 TO n DO
          BEGIN
            g[i] := (x1 - a[i]) * g[i-1] - b[i] * g[i-2];

```

```

      g1[i] := g1[i - 1] + c[i] * g[i];
    END;
    d := d + (y[k]-g1[n])*(y[k]-g1[n]);
    WRITELN ('x = ',x1:6:3,'   Q',n:1,' = ',g1[n]:6:3);
  END;
    d := SQRT(d / y2);
    WRITELN ('Погрешность приближения d = ', d:8:6);
    PAUSA;
  END.

```

4¹. Программа вычисления
 многочленов наилучшего среднеквадратичного
 приближения функций ортогональными многочленами
 на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
      program ApproxByPolinomsOfTchebishev
C      ***** РАСЧЕТ *****
C      ТАБЛИЦЫ ЗНАЧЕНИЙ СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
C      ФУНКЦИИ ОРТОГОНАЛЬНЫМИ МНОГОЧЛЕНАМИ ЧЕБЫШЕВА
C      *****
      INTEGER*2 system
      COMMON m,z,g1(0:15)/b1/y(15),c(0:14)
      COMMON /b2/x(15),g(0:15),a(2:14),b(2:14)
      i=system('cls')
      WRITE (*,*) 'Введите m - число значений функции (m < 16)'
      READ (*,*) m
      WRITE (*,*) 'Введите n - порядок многочлена'
      READ (*,*) n
      WRITE (*,*) 'Вводите пары значений x1,y1; i=1,...,m'
      a0 = 0
      y1 = 0
      y2 = 0
      DO i=1,m
        READ (*,*) x(1),y(1)
        WRITE (*,*)
        a0 = a0 + x(1)
        y1 = y1 + y(1)
        y2 = y2 + y(1)**2
      END DO
      a0 = a0 / m

```

```

      c(0) = y1 / m
      DO i = 1,m
        g(1) = 1
        g1(1) = x(1) - a0
      END DO
      IF (n.NE.0) THEN
        IF (n.GT.m - 1) n = m - 1
        DO i = 2,n
          CALL CalculOfCoeffOfFourie (1)
          CALL CalculOfParameters (1)
          DO j = 1,m
            a1 = g1(j)
            g1(j) = (x(j) - a(1)) * g1(j) - b(1) * g(j)
            g(j) = a1
          END DO
        END DO
        CALL CalculOfCoeffOfFourie (n+1)
      END IF

      d = 0
      DO k = 1,m
        x1 = x(k)
        g(0) = 1
        g(1) = x1 - a0
        g1(0) = c(0)
        g1(1) = g1(0) + c(1) * g(1)
        DO i = 2,n
          g(i) = (x1 - a(1)) * g(i - 1) - b(1) * g(i - 2)
          g1(i) = g1(i - 1) + c(1) * g(i)
        END DO
        d = d + (y(k) - g1(n))**2
        WRITE (*, '(A4,F5.2\)' ) ' x =', x1
        WRITE (*, '(A6,I1,A2,F6.3)' ) '      Q', n, '= ', g1(n)
      END DO
      d = SQRT(d / y2)
      WRITE (*, '(A28,F8.6)' ) ' Погрешность приближения d =', d
      PAUSE 'Нажмите клавишу ENTER для продолжения...'
      END

```

C

ПОДПРОГРАММЫ

C

C

Вычисление коэффициентов Фурье $c(k)$

SUBROUTINE CalculOfCoeffOfFourie (1)

COMMON m,z,g1(0:15)/b1/y(15),c(0:14)

z = 0

c1 = 0

DO j = 1, m

c1 = c1 + y(j) * g1(j)

z = z + g1(j) * g1(j)

END DO

c(1 - 1) = c1 / z

RETURN

END

C

C

Вычисление параметров рекуррентной формулы $a(k)$, $b(k)$

SUBROUTINE CalculOfParameters (1)

REAL k0,n0

COMMON m,z,g1(0:15)/b2/x(15),g(0:15),a(2:14),b(2:14)

k0 = 0

b0 = 0

n0 = 0

DO j = 1, m

k0 = k0 + x(j) * g1(j) * g1(j)

b0 = b0 + x(j) * g(j) * g1(j)

n0 = n0 + g(j) * g(j)

END DO

a(1) = k0 / z

b(1) = b0 / n0

RETURN

END

5¹. Программа вычисления

многочленов наилучшего среднеквадратичного
 приближения функций ортогональными многочленами
 на языке C

/*

***** РАСЧЕТ *****

ТАБЛИЦЫ ЗНАЧЕНИЙ СРЕДНЕКВАДРАТИЧНОГО ПРИБЛИЖЕНИЯ
 ФУНКЦИИ ОРТОГОНАЛЬНЫМИ МНОГОЧЛЕНАМИ ЧЕБЫШЕВА

```

*****/
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define L 15
int i,m;
float a[L],b[L],c[L],n1,x[L+1],y[L+1],g[L+1],g[L+1];
/* -----
                                ПОДПРОГРАММЫ
        Вычисление коэффициентов Фурье c[k] */
void calculofcoeffoffourie(i)
int i;
{ int j;
  float c1;
  for (n1=c1=0,j=1; j <= m; j++)
    { c1 += y[j] * g1[j]; n1 += g1[j] * g1[j]; };
  c[i - 1] = c1 / n1;
};
/* Вычисление параметров рекуррентной формулы a[k], b[k] */
void calculofparameters(i)
int i;
{ int j;
  float k0,b0,n0;
  for (k0=b0=n0=0,j=1; j <= m; j++)
    { k0 += x[j] * g1[j] * g1[j]; b0 += x[j] * g[j] * g1[j];
      n0 += g[j] * g[j]; };
  a[i] = k0 / n1; b[i] = b0 / n0;
}
/* -----
                                ОСНОВНАЯ ПРОГРАММА */
void main()
{ int j,k,n;
  float a0,a1,d,x1,y1,y2;
  clrscr();
  printf ("Введите m - число значений функции (m < 16)\n");
  scanf ("%i",&m);
  printf ("Введите n - порядок аппроксимирующего многочлена n");
  scanf ("%i",&n);

```

```

printf ("Вводите пары значений x1,y1; i=1,...,m\n");
for (a0=y1=y2=0,i=1; i <= m; i++,printf("\n"))
{ scanf ("%f%f",&x[i],&y[i]);
  a0 += x[i]; y1 += y[i]; y2 += y[i] * y[i];
};
c[0] = y1 / m;
for (a0 /= m,i=1; i <= m; i++)
{ g[i] = 1; g1[i] = x[i] - a0; };
if (n != 0)
{ if (n > m - 1) n = m - 1;
  for (i = 2; i <= n; i++)
  { calculofcoeffoffourie(i); calculofparameters(i);
    for (j = 1; j <= m; j++)
    { a1 = g1[j];
      g1[j] = (x[j] - a[i]) * g1[j] - b[i] * g[j];
      g[j] = a1;
    };
  }; calculofcoeffoffourie(n+1);
};
for (d=0,k=1; k <= m; k++)
{ x1 = x[k];
  g[0] = 1; g[1] = x1 - a0;
  g1[0] = c[0]; g1[1] = g1[0] + c[1] * g[1];
  for (i = 2; i <= n; i++)
  { g[i] = (x1 - a[i]) * g[i - 1] - b[i] * g[i - 2];
    g1[i] = g1[i - 1] + c[i] * g[i];
  };
  d += (y[k]-g1[n])*(y[k]-g1[n]);
  printf ("x = %f   Q%i = %f\n",x1,n,g1[n]);
};
d = sqrt(d / y2);
printf ("Погрешность приближения d = %f",d);
printf ("\nНажмите любую клавишу для продолжения...");
getch();
}

```


Примечание. В случае отрицательных значений x_i , y_i требуется предварительное преобразование исходных данных. Если, например, $x_0 = \min_{1 \leq i \leq n} \{x_i\} < 0$ и $y_0 = \min_{1 \leq i \leq n} \{y_i\} < 0$, то программу можно применять для переменных $\tilde{x}_i = |x_0| + 1 + x_i$, $\tilde{y}_i = |y_0| + 1 + y_i$.

**2⁰. Программа вычисления параметров
эмпирической формулы с двумя параметрами
методом наименьших квадратов на языке BASIC**

```

1 REM *****
2 REM      ОПРЕДЕЛЕНИЕ ПАРАМЕТРОВ ЭМПИРИЧЕСКОЙ ФОРМУЛЫ
3 REM      С ДВУМЯ ПАРАМЕТРАМИ
4 REM      МЕТОДОМ НАИМЕНЬШИХ КВАДРАТОВ
5 REM *****
10 DIM x(15), y(15), x1(15), y1(15), a(6), b(6), d(6)
20 DEFINT I-K, M: CLS
30 INPUT "Введите число экспериментальных точек m > 1 "; m
40 PRINT " Вводите через запятую пары значений "
41 PRINT "      x1,y1; i=1,...,m"
50 FOR i = 1 TO m
60   INPUT x(i), y(i)
70   x1(i) = x(i): y1(i) = y(i)
80 NEXT i
90 FOR j = 0 TO 6
100  ON j GOSUB 340, 370, 400, 430, 460, 490
110  GOSUB 200
120 NEXT j
130 d = d(0): k = 0
140 FOR i = 1 TO 6
150   IF d(i) < d THEN d = d(i): k = i
160 NEXT i
161 a$ = "Эмпирическая формула N#: d=##.#####; "
162 b$ = "      k=###.###^ ^ ^ ^;   b=###.###^ ^ ^ ^"
170 PRINT USING a$; k; d;
180 PRINT USING b$; a(k); b(k)
190 GOSUB 520: END
200 a1 = 0: b1 = 0: a2 = 0: b2 = 0
210 FOR i = 1 TO m

```

```

220  a1 = a1 + x1(1): b1 = b1 + y1(1)
230  a2 = a2 + x1(1) ^ 2: b2 = b2 + x1(1) * y1(1)
240  NEXT 1
250  d = m * a2 - a1 ^ 2
260  k! = (m * b2 - a1 * b1) / d: b = (b1 * a2 - a1 * b2) / d
270  d = 0: f2 = 0
280  FOR 1 = 1 TO m
290    f2 = f2 + y1(1) ^ 2: d = d + (y1(1) - k! * x1(1) - b) ^ 2
300  NEXT 1
310  d(j) = SQR(d / f2)
320  a(j) = k!: b(j) = b
330  RETURN
340  FOR 1 = 1 TO m
350    y1(1) = x(1) * y(1)
360  NEXT 1: RETURN
370  FOR 1 = 1 TO m
380    y1(1) = 1 / y(1)
390  NEXT 1: RETURN
400  FOR 1 = 1 TO m
410    y1(1) = x(1) / y(1)
420  NEXT 1: RETURN
430  FOR 1 = 1 TO m
440    y1(1) = LOG(y(1))
450  NEXT: RETURN
460  FOR 1 = 1 TO m
470    x1(1) = LOG(x(1)): y1(1) = y(1)
480  NEXT 1: RETURN
490  FOR 1 = 1 TO m
500    y1(1) = LOG(y(1))
510  NEXT 1: RETURN
520 PRINT : PRINT "Для продолжения нажмите любую клавишу"
530 IF INKEY$ = "" THEN 530
540 RETURN

```

**3⁰. Программа вычисления параметров
эмпирической формулы с двумя параметрами
методом наименьших квадратов на языке PASCAL**

```
program MethodOfLeastSquares;
```

```

{ *****
      ОПРЕДЕЛЕНИЕ ПАРАМЕТРОВ ЭМПИРИЧЕСКОЙ ФОРМУЛЫ
      С ДВУМЯ ПАРАМЕТРАМИ
      МЕТОДОМ НАИМЕНЬШИХ КВАДРАТОВ
      *****}

uses Crt;
const
  l=15;
type
  parameters=array [0..6] of real;
  data=array[1..l] of real;
var
  i,j,k,m:integer;
  d1:real;
  a,b,d:parameters;
  x,y,x1,y1:data;
  ch:char;
{ -----
      ПОДПРОГРАММЫ
      -----}

procedure PAUSA;
BEGIN
  WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
  REPEAT ch := readkey UNTIL ch <> '';
END;
{ -----
      Вычисление параметров линейной функции при выравнивании
      экспериментальных данных
      -----}
procedure calculOfParameters ;
var
  k0,b0,a1,b1,a2,b2,f2:real;
BEGIN
  a1 := 0; b1 := 0; a2 := 0; b2 := 0;
  FOR i := 1 TO m DO
    BEGIN
      a1 := a1 + x1[i]; b1 := b1 + y1[i];
      a2 := a2 + x1[i] * x1[i]; b2 := b2 + x1[i] * y1[i];
    END;

```

```

d1 := m * a2 - a1 * a1;
k0 := (m * b2 - a1 * b1)/d1; b0 := (b1 * a2 - a1 * b2)/d1;
d1 := 0; f2 := 0;
FOR 1 := 1 TO m DO
  BEGIN
    f2 := f2 + y1[1] * y1[1];
    d1 := d1 + (y1[1]-k0*x1[1]-b0) * (y1[1]-k0*x1[1]-b0);
  END;
d[j] := SQRT(d1 / f2);
a[j] := k0; b[j] := b0;
END;
{ -----

```

ОСНОВНАЯ

ПРОГРАММА

}

```

BEGIN
  ClrScr;
  WRITELN ('Введите число экспериментальных точек m > 1');
  READ (m);
  WRITELN ('Введите пары значений x1,y1; 1=1,...,m');
  FOR 1 := 1 TO m DO
    BEGIN
      READ (x[1],y[1]);
      WRITELN;
    END;
  FOR j := 0 TO 6 DO
    BEGIN
      CASE j OF
        0: FOR 1 := 1 TO m DO
            BEGIN
              x1[1] := x[1]; y1[1] := y[1];
            END;
        1: FOR 1 := 1 TO m DO y1[1] := x[1] * y[1];
        2: FOR 1 := 1 TO m DO y1[1] := 1 / y[1];
        3: FOR 1 := 1 TO m DO y1[1] := x[1] / y[1];
        4: FOR 1 := 1 TO m DO y1[1] := LN(y[1]);
        5: FOR 1 := 1 TO m DO
            BEGIN
              x1[1] := LN(x[1]); y1[1] := y[1];
            END;

```

```

6: FOR 1 := 1 TO m DO y[1] := LN(y[1]);
END;
  CalculOfParameters;
END;
  d1 := d[0]; k := 0;
  FOR 1 := 1 TO 6 DO
    IF d[1] < d1 THEN
      BEGIN
        d1 := d[1]; k := 1;
      END;
    WRITE ('Эмпирическая формула N',k:1,' d=',d1:8:5);
    WRITELN (' k=',a[k]:8:5,' b=',b[k]:8:5);
  PAUSA;
END.

```

4⁰. Программа вычисления параметров эмпирической формулы с двумя параметрами методом наименьших квадратов на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
  program MethodOfLeastSquares
C      *****
C      ОПРЕДЕЛЕНИЕ ПАРАМЕТРОВ ЭМПИРИЧЕСКОЙ ФОРМУЛЫ
C      С ДВУМЯ ПАРАМЕТРАМИ
C      МЕТОДОМ НАИМЕНЬШИХ КВАДРАТОВ
C      *****
    INTEGER*2 system
    DIMENSION x(15),y(15)
    COMMON m,j,x1(15),y1(15),a(0:6),b(0:6),d(0:6)
    i=system('cls'C)
    WRITE (*,*) 'Введите число экспериментальных точек m > 1'
    READ (*,*) m
    WRITE (*,*) 'Введите пары значений x1,y1; i=1,...,m'
    DO i=1,m
      READ (*,*) x(1),y(1)
      WRITE (*,*)
    END DO
  END

```

```
END DO
DO j = 0,6
  SELECT CASE (j)
    CASE (0)
      DO i = 1,m
        x1(i) = x(i)
        y1(i) = y(i)
      END DO
    CASE (1)
      DO i = 1,m
        y1(i) = x(i) * y(i)
      END DO
    CASE (2)
      DO i = 1,m
        y1(i) = 1 / y(i)
      END DO
    CASE (3)
      DO i = 1,m
        y1(i) = x(i) / y(i)
      END DO
    CASE (4)
      DO i = 1,m
        y1(i) = LOG(y(i))
      END DO
    CASE (5)
      DO i = 1,m
        x1(i) = LOG(x(i))
        y1(i) = y(i)
      END DO
    CASE (6)
      DO i = 1,m
        y1(i) = LOG(y(i))
      END DO
  END SELECT
  CALL CalculOfParameters
END DO
d1 = d(0)
k = 0
```

```
DO 1 = 1, 6
```

```
  IF (d(1).LT.d1) THEN
```

```
    d1 = d(1)
```

```
    k = 1
```

```
  END IF
```

```
END DO
```

```
WRITE (*, '(A23,I1)') 'Эмпирическая формула N',k
```

```
WRITE (*, '(A3,E12.6)') ' d=',d(k)
```

```
WRITE (*, '(A3,E12.6,A4,E12.6)') ' k=',a(k), ' b=',b(k)
```

```
PAUSE 'Нажмите клавишу ENTER для продолжения...'
```

```
END
```

```
C -----
C  Вычисление параметров линейной функции при выравнивании
C  экспериментальных данных
```

```
SUBROUTINE calculoOfParameters
```

```
REAL k0
```

```
COMMON m,j,x1(15),y1(15),a(0:6),b(0:6),d(0:6)
```

```
  a1 = 0
```

```
  b1 = 0
```

```
  a2 = 0
```

```
  b2 = 0
```

```
DO 1 = 1,m
```

```
  a1 = a1 + x1(1)
```

```
  b1 = b1 + y1(1)
```

```
  a2 = a2 + x1(1)*x1(1)
```

```
  b2 = b2 + x1(1)*y1(1)
```

```
END DO
```

```
  d1 = m * a2 - a1 * a1
```

```
  k0 = (m * b2 - a1 * b1) / d1
```

```
  b0 = (b1 * a2 - a1 * b2) / d1
```

```
  d1 = 0
```

```
  f2 = 0
```

```
DO 1 = 1,m
```

```
  f2 = f2 + y1(1) * y1(1)
```

```
  d1=d1 + (y1(1) - k0*x1(1)-b0) * (y1(1) - k0*x1(1)-b0)
```

```
END DO
```

```
  d(j) = SQRT(d1 / f2)
```

```
  a(j) = k0
```

```

b(j) = b0
RETURN
END

```

**5⁰. Программа вычисления параметров
эмпирической формулы с двумя параметрами
методом наименьших квадратов на языке C**

```

/* *****
ОПРЕДЕЛЕНИЕ ПАРАМЕТРОВ ЭМПИРИЧЕСКОЙ ФОРМУЛЫ
С ДВУМЯ ПАРАМЕТРАМИ
МЕТОДОМ НАИМЕНЬШИХ КВАДРАТОВ
***** */

#include <stdio.h>
#include <conio.h>
#include <math.h>
#define L 15
int i,j,k,m;
float d1,a[7],b[7],d[7],x[L+1],y[L+1],x1[L+1],y1[L+1];
/* -----
ПОДПРОГРАММА
Вычисление параметров линейной функции при выравнивании
экспериментальных данных */
void calculofparameters(void)
{ float k0,b0,a1,b1,a2,b2,f2;
  for (a1=b1=a2=b2=0,i=1; i <= m; i++)
    { a1 = a1 + x1[i]; b1 = b1 + y1[i];
      a2 = a2 + x1[i] * x1[i]; b2 = b2 + x1[i] * y1[i];
    };
  d1 = m * a2 - a1 * a1;
  k0 = (m * b2 - a1 * b1)/d1; b0 = (b1 * a2 - a1 * b2)/d1;
  for (d1=f2=0,i=1; i <= m; i++)
    { f2 = f2 + y1[i] * y1[i];
      d1 = d1 + (y1[i]-k0*x1[i]-b0) * (y1[i]-k0*x1[i]-b0);
    };
  d[j] = sqrt(d1 / f2);
  a[j] = k0; b[j] = b0;
}

```



```

}
/* -----
                                ОСНОВНАЯ      ПРОГРАММА                                */
void main()
{ clrscr();
  printf ("Введите число экспериментальных точек m > 1\n");
  scanf ("%i",&m);
  printf ("Вводите пары значений x1,y1; 1=1,...,m\n");
  for(i=1; i<=m; i++,printf("\n")) scanf ("%i%i",&x[i],&y[i]);
  for (j = 0; j <= 6; j++)
  {
    switch (j) {
      case 0:  for (i = 1; i <= m; i++)
                { x[i] = x[i]; y[i] = y[i]; };          break;
      case 1:  for (i = 1; i <= m; i++) y[i]=x[i] * y[i]; break;
      case 2:  for (i = 1; i <= m; i++) y[i] = i / y[i]; break;
      case 3:  for (i = 1; i <= m; i++) y[i]=x[i]/y[i];  break;
      case 4:  for (i = 1; i <= m; i++) y[i]=log(y[i]);  break;
      case 5:  for (i = 1; i <= m; i++)
                { x[i] = log(x[i]); y[i] = y[i]; };      break;
      case 6:  for (i = 1; i <= m; i++) y[i] = log(y[i]); break;
    }
    calculofparameters();
  };

  for (d1=d[0],k=0,i=1; i <= 6; i++)
    if (d[i] < d1) { d1 = d[i]; k = i; };
  printf ("Эмпирическая формула N%i d=%i",k,d1);
  printf (" k=%i b=%i\n",a[k],b[k]);
  printf ("\nНажмите любую клавишу для продолжения...");
  getch();
}

```

**6⁰. Программа вычисления параметров
эмпирической формулы с двумя параметрами
методом наименьших квадратов на языке QUICKBASIC**

Программа позволяет определить эмпирическую формулу с двумя параметрами, наилучшим образом аппроксимирующую заданную таблицей функцию (экспериментальную зависимость), при любом числе

пар $(x(i), y(i))$ ($1 \leq i \leq m$), допускаемом памятью компьютера, используя метод наименьших квадратов. Предполагается, что значения $x(i)$ и $y(i)$ положительны, а величины $x(i)$ упорядочены по возрастанию, т. е. $0 < x(1) < \dots < x(m)$.

В результате выполнения программы на экране дисплея появляются одна из шести формул (см. таблицу на с.85), график функции, соответствующий полученной формуле, и точки, отвечающие экспериментальным данным.

```

' *****
'      ОПРЕДЕЛЕНИЕ ЭМПИРИЧЕСКОЙ ФОРМУЛЫ
'      С ДВУМЯ ПАРАМЕТРАМИ
'      МЕТОДОМ НАИМЕНЬШИХ КВАДРАТОВ
'      С ГРАФИЧЕСКОЙ ИЛЛЮСТРАЦИЕЙ
' *****
DECLARE SUB choice (j%)
DECLARE SUB calculOfparameters ()
DECLARE FUNCTION f! (variant%, x!, k!, b!)
DEFINT I-K, M: DIM a(6), b(6), d(6): CLS
  INPUT "Введите число экспериментальных точек m > 1 "; m
  DIM x(m), y(m), x1(m), y1(m)
  PRINT " Вводите через запятую пары значений "
  PRINT "      x1,y1; i=1,...,m"
  FOR i = 1 TO m
    INPUT x(i), y(i)
  NEXT
  FOR j = 0 TO 6
    CALL choice(j)
    CALL calculOfparameters
  NEXT
  'Выбор формулы, приближающей наилучшим образом эксперименталь-
  'ные данные по условию d=d_minimum
  '-----
      d = d(0): k = 0
  FOR i = 1 TO 6
    IF d(i) < d THEN d = d(i): k = i
  NEXT
  NEXT

```

Определение границ экспериментальных значений функции для

```

      miny! = y(1): maxy! = y(1)
FOR 1 = 2 TO m
      IF y(1) < miny! THEN miny! = y(1)
      IF y(1) > maxy! THEN maxy! = y(1)
NEXT
      IF miny! = maxy! THEN
            miny! = miny! - ABS(miny!) / 2
            maxy! = maxy! + ABS(maxy!) / 2
      END IF

```

```

SCREEN 9
VIEW (10, 20)-(630, 300), 1
WINDOW (x(1), miny!)-(x(m), maxy!)
h = (x(m) - x(1)) / 100
x = x(1)
PSET (x, f(k, x, a(k), b(k)))
FOR 1 = 1 TO 100
      x = x + h
      LINE -(x, f(k, x, a(k), b(k)))
NEXT
r = (x(m) - x(1)) / 200
FOR 1 = 1 TO m
      CIRCLE (x(1), y(1)), r, 4
      PAINT (x(1), y(1)), 4
NEXT
LOCATE 1, 3: PRINT "Эмпирическая формула N"; k; ": ";
SELECT CASE k
CASE 0: PRINT "y = ("; a(k); ") x + ("; b(k); ")"
CASE 1: PRINT "y = ("; a(k); ") + ("; b(k); ")/x "
CASE 2: PRINT "y = 1/(("; a(k); ") x + ("; b(k); "))."
CASE 3: PRINT "y = x/(("; a(k); ") x + ("; b(k); "))."
CASE 4: PRINT "y = ("; EXP(b(k)); ")*("; EXP(a(k)); ")^x"
CASE 5: PRINT "y = ("; a(k); ")*Ln x + ("; b(k); ")"
CASE 6: PRINT "y = ("; EXP(b(k)); ")*"; "x^("; a(k); ")"
END SELECT

```

```

LOCATE 23, 40
PRINT "Для продолжения нажмите любую клавишу"
a$ = INPUT$(1)
END

DEFSNG I-K, M
SUB calculOfparameters
DEFINT I-J, M: SHARED j, m, x1(), y1(), a(), b(), d()
      a1 = 0: b1 = 0: a2 = 0: b2 = 0
FOR i = 1 TO m
  a1 = a1 + x1(i): b1 = b1 + y1(i)
  a2 = a2 + x1(i) ^ 2: b2 = b2 + x1(i) * y1(i)
NEXT
  d = m * a2 - a1 ^ 2
  k = (m * b2 - a1 * b1) / d: b = (b1 * a2 - a1 * b2) / d
  d = 0: f2 = 0
FOR i = 1 TO m
  f2 = f2 + y1(i) ^ 2: d = d + (y1(i) - k * x1(i) - b) ^ 2
NEXT
  a(j) = k: b(j) = b: d(j) = SQR(d / f2)
END SUB

DEFSNG I-J, M
SUB choice (j%)
DEFINT I-J, M: SHARED m, x(), y(), x1(), y1()
SELECT CASE j
CASE 0
  FOR i = 1 TO m
    x1(i) = x(i): y1(i) = y(i)
  NEXT
CASE 1
  FOR i = 1 TO m
    y1(i) = x(i) * y(i)
  NEXT
CASE 2
  FOR i = 1 TO m

```

```

                                 $y1(1) = 1 / y(1)$ 
NEXT
CASE 3
  FOR 1 = 1 TO m
                                 $y1(1) = x(1) / y(1)$ 
  NEXT
CASE 4
  FOR 1 = 1 TO m
                                 $y1(1) = \text{LOG}(y(1))$ 
  NEXT
CASE 5
  FOR 1 = 1 TO m
                                 $x1(1) = \text{LOG}(x(1))$ ;  $y1(1) = y(1)$ 
  NEXT
CASE 6
  FOR 1 = 1 TO m
                                 $y1(1) = \text{LOG}(y(1))$ 
  NEXT
END SELECT
END SUB

DEFSNG I-J, M
FUNCTION f (variant%, x, k, b)
DEFINT V
  SELECT CASE variant
    CASE 0:  $f = k * x + b$ 
    CASE 1:  $f = k + b / x$ 
    CASE 2:  $f = 1 / (k * x + b)$ 
    CASE 3:  $f = x / (k * x + b)$ 
    CASE 4:  $a = \text{EXP}(b)$ ;  $c = \text{EXP}(k)$ ;  $f = a * c ^ x$ 
    CASE 5:  $f = k * \text{LOG}(x) + b$ 
    CASE 6:  $a = \text{EXP}(b)$ ;  $f = a * x ^ k$ 
  END SELECT
END FUNCTION

```

1⁰. Блок-схема построения интерполяционного многочлена Лагранжа

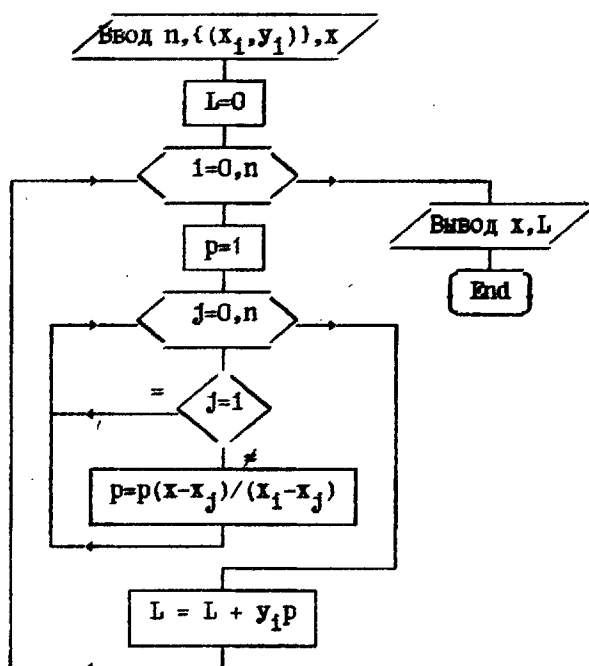
По определенным значениям y_i некоторой функции в точках x_i ($i=0,1,2,\dots,n$) требуется составить программу вычисления значения многочлена Лагранжа $L(x)$ в точке $x \in [x_0, x_n]$.

Для построения многочлена используются рекуррентные соотношения

$$p_{i0}(x)=1, \quad p_{ij}(x)=p_{i(j-1)}(x) \frac{x-x_j}{x_i-x_j}, \quad j \neq i$$

$$(j = 0, 1, 2, \dots, i-1, i+1, \dots, n);$$

$$L_0(x) = 0, \quad L_{i+1}(x) = L_i(x) + y_i p_{in} \quad (i=0, 1, 2, \dots, n).$$



2⁰. Программа вычисления значения многочлена Лагранжа на языке BASIC

```

1 REM *****
2 REM      ВЫЧИСЛЕНИЕ ЗНАЧЕНИЯ МНОГОЧЛЕНА ЛАГРАНЖА
3 REM      ДЛЯ ФУНКЦИИ, ОПРЕДЕЛЕННОЙ
4 REM      В n+1 УЗЛЕ ИНТЕРПОЛЯЦИИ x(i) (i=0,1,...,n)
5 REM *****
10 DEFINT I-J, N: DIM x(15), y(15)
20 CLS
30 INPUT "Введите n - порядок многочлена Лагранжа "; n
40 PRINT "Введите через запятую пары значений xi,yi; i=0,...,n"
50 FOR i = 0 TO n
60   INPUT x(i), y(i)
70 NEXT i
80 INPUT "Введите x "; x
90 L = 0
100 FOR i = 0 TO n
110   p = 1
120   FOR j = 0 TO n
130     IF j <> i THEN p = p * (x - x(j)) / (x(i) - x(j))
140   NEXT j
150   L = L + y(i) * p
160 NEXT i
170 PRINT USING "x = ###.### L(x) = ###.###"; x, L
180 GOSUB 190: END
190 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
200 IF INKEY$ = "" THEN 200
210 RETURN

```

3⁰. Программа вычисления значения многочлена Лагранжа на языке PASCAL

```

program PolinomOfLagrange;
(
  *****
  ВЫЧИСЛЕНИЕ ЗНАЧЕНИЯ МНОГОЧЛЕНА ЛАГРАНЖА
  ДЛЯ ФУНКЦИИ, ОПРЕДЕЛЕННОЙ
  В n+1 УЗЛЕ ИНТЕРПОЛЯЦИИ x(i) (i=0,1,...,n)
  *****
)
uses Crt;

```

```

const
    m = 15;
type
    vector=array[0..m] of real;
var
    i,j,n:integer;
    p,L,x1:real;
    x,y:vector;
    ch:char;
{ -----
                                ПОДПРОГРАММА                                }
procedure PAUSA;
begin
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
    END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                }
begin
    clrscr;
    WRITELN ('Введите n - порядок многочлена Лагранжа');
    READ (n);
    WRITELN ('Введите пары значений x1,y1; i=0,...,n');
    FOR i := 0 TO n DO
        BEGIN
            READ (x[i], y[i]);
            WRITELN;
        END;
    WRITELN ('Введите x');
    READ (x1);
    L := 0;
    FOR i := 0 TO n DO
        BEGIN
            p := 1;
            FOR j := 0 TO n DO
                IF j <> i THEN p := p * (x1 - x[j]) / (x[i] - x[j]);
            L := L + y[i] * p;
        END;
    END;

```



```
WRITELN ('x = ',x1:8:4,'    L(x) = ',L:8:4);
```

```
PAUSA;
```

```
END.
```

4⁰. Программа вычисления значения многочлена Лагранжа на языке **FORTRAN**

```
$INCLUDE: 'EXEC.FI'
```

```
program PolinomOfLagrange
```

```
C *****
```

```
C      ВЫЧИСЛЕНИЕ ЗНАЧЕНИЯ МНОГОЧЛЕНА ЛАГРАНЖА
```

```
C      ДЛЯ ФУНКЦИИ, ОПРЕДЕЛЕННОЙ
```

```
C      В n+1 УЗЛЕ ИНТЕРПОЛЯЦИИ x(i) (i=0,1,...,n)
```

```
C *****
```

```
REAL l
```

```
INTEGER*2 system
```

```
DIMENSION x(0:15),y(0:15)
```

```
i=system('cls'C)
```

```
WRITE(*,*) 'Введите n - порядок многочлена Лагранжа'
```

```
READ (*,*) n
```

```
WRITE(*,*) 'Введите пары значений x(i),y(i); i=0,1,...,n'
```

```
DO i = 0, n
```

```
    READ (*,*) x(i),y(i)
```

```
    WRITE (*,*)
```

```
END DO
```

```
WRITE(*,*) 'Введите x '
```

```
READ (*,*) x1
```

```
L = 0
```

```
DO i = 0,n
```

```
    p = 1
```

```
    DO j = 0,n
```

```
        IF (j.NE.i) p = p * (x1 - x(j)) / (x(i) - x(j))
```

```
    END DO
```

```
    L = L + y(i) * p
```

```
END DO
```

```
WRITE (*, '(A5,F8.4,A10,F8.4)') ' x = ',x1,'    L(x) = ', L
```

```
PAUSE 'Для продолжения нажмите клавишу ENTER...'
```

```
END
```

5⁰. Программа вычисления значения многочлена Лагранжа на языке C

```

/* *****
   ВЫЧИСЛЕНИЕ ЗНАЧЕНИЯ МНОГОЧЛЕНА ЛАГРАНЖА
   ДЛЯ ФУНКЦИИ, ОПРЕДЕЛЕННОЙ
   В (n + 1) УЗЛЕ ИНТЕРПОЛЯЦИИ x(i) (i=0,1,...,n)
   ***** */

#include <stdio.h>
#include <conio.h>
#define M 15
/* ----- */

void main()
{ int i,j,n;
  float p,L,x1,x[M+1],y[M+1];
  clrscr();
  printf ("Введите n - порядок многочлена Лагранжа n");
  scanf ("%i",&n);
  printf ("Вводите пары значений x1,y1; i=0,...,n\n");
  for (i = 0; i <= n; i++,printf("\n"))
    scanf ("%f%f",&x[i], &y[i]);
  printf ("Введите x\n");
  scanf ("%f",&x1);
  for (L=0,i=0; i <= n; i++)
    for (p=1,j=0; j <= n; j++)
      if (j != i) p=p*(x1-x[j])/(x[i]-x[j]);
      L = L + y[i] * p;};
  printf ("\nx = %f   L(x) = %f",x1,L);
  printf ("\nНажмите любую клавишу для продолжения...");
  getch();
}

```

К §2

1⁰. Блок-схема построения кубического сплайна

Пусть отрезок $[a,b]$ разбит на n равных частей и в точках x_i ($i=0,1,2,\dots,n$; $x_0=a$, $x_n=b$) некоторая функция принимает значения

y_i . Для переменной x , принадлежащей части разбиения $[x_{i-1}, x_i]$ ($i=1, \dots, n$), определена функция (кубический многочлен)

$$S_i(x) = y_{i-1} \frac{(x-x_i)^2(2(x-x_{i-1})+h)}{h^3} + y_i \frac{(x-x_{i-1})^2(2(x_i-x)+h)}{h^3} + \\ + m_{i-1} \frac{(x-x_i)^2(x-x_{i-1})}{h^2} + m_i \frac{(x-x_{i-1})^2(x-x_i)}{h^2}.$$

Здесь $h = \frac{b-a}{n}$ — шаг разбиения отрезка. Неизвестные m_i определяются рекуррентными соотношениями

$$m_0 = A; \quad m_n = B; \quad m_i = L_i m_{i+1} + M_i \quad (i=n-1, n-2, \dots, 0)$$

после предварительного вычисления вспомогательных величин M_i , L_i по рекуррентным формулам

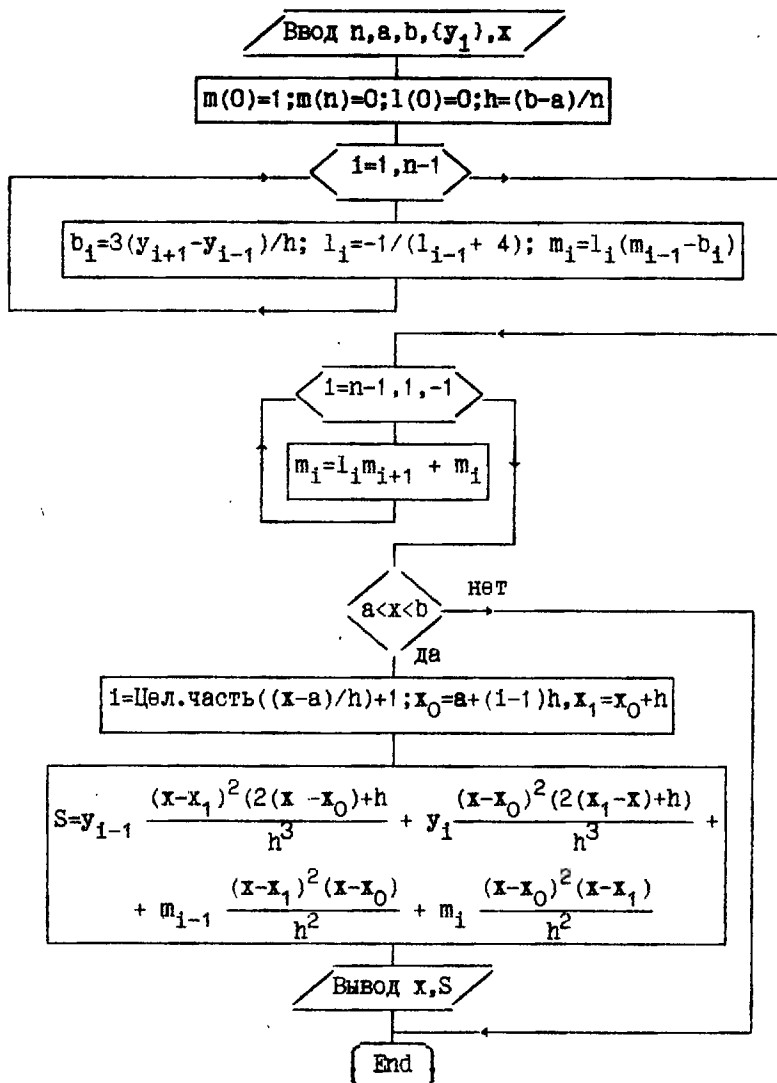
$$L_0=0, \quad M_0=m_0, \quad L_i = \frac{-1}{L_{i-1} + 4}, \quad M_i = L_i(M_{i-1} - b_i) \quad (i=1, 2, \dots, n-1),$$

где
$$b_i = \frac{3(y_{i+1} - y_{i-1})}{h}.$$

Величины A и B должны быть заданы. При построении кубического сплайна, интерполирующего дифференцируемую функцию $y=f(x)$ по системе точек, полагают $A=f'(a)$, $B=f'(b)$ (краевые условия I типа). Выбор необходимой формулы $S_i(x)$ для заданного значения переменной x определяется целым числом i :

$$i = \text{целая часть} \left[\frac{x-a}{h} \right] + 1.$$

Работа программ проверяется на примерах 1 и 2 из §2. В соответствии с условиями задач в программах принято $m_0=1$, $m_n=0$.



2⁰. Программа построения кубического сплайна,
интерполирующего заданную функцию,
на языке BASIC

1 REM *****
2 REM ИНТЕРПОЛИРОВАНИЕ ФУНКЦИИ С ПОМОЩЬЮ КУБИЧЕСКОГО СПЛАЙНА

```

3 REM *****
10 DEFINT I, N: DIM y(15), b(14), m(15), l(14)
20 CLS
30 INPUT "Введите n - число разбиений отрезка [a,b] "; n
40 PRINT "Введите через запятую координаты концов отрезка a, b"
50 INPUT a, b
60 PRINT "Введите значения функции y(i) в узлах интерполяции"
70 PRINT "(i = 0, 1, 2, ..., n), причем y(0) = f(a), y(n) = f(b)"
80 FOR i = 0 TO n
90 INPUT y(i)
100 NEXT i
110 INPUT "Введите значение x "; x
111 REM Условия на концах отрезка: m(0)=f'(a)=1 и m(n)=f'(b)=0
120 m(0) = 1: m(n) = 0: l(0) = 0: h = (b - a) / n
130 FOR i = 1 TO n - 1
140 b(i) = 3 * (y(i + 1) - y(i - 1)) / h
150 l(i) = -1 / (l(i - 1) + 4)
160 m(i) = l(i) * (m(i - 1) - b(i))
170 NEXT i
180 FOR i = n - 1 TO 1 STEP -1
190 m(i) = l(i) * m(i + 1) + m(i)
200 NEXT i
210 IF x <= a OR x >= b THEN 280
220 i = INT((x - a) / h) + 1: x0 = a + (i - 1) * h: x1 = x0 + h
230 s = y(i - 1) * ((x - x1) ^ 2 * (2 * (x - x0) + h)) / h ^ 3
240 s = s + y(i) * ((x - x0) ^ 2 * (2 * (x1 - x) + h)) / h ^ 3
250 s = s + m(i - 1) * (x - x1) ^ 2 * (x - x0) / h ^ 2
260 s = s + m(i) * (x - x0) ^ 2 * (x1 - x) / h ^ 2
270 PRINT USING "x = ##.####^#### s = ##.#####^####"; x; s
280 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
290 IF INKEY$ = "" THEN 290
300 END

```

3⁰. Программа построения кубического сплайна, интерполирующего заданную функцию,

на языке PASCAL

```

{ *****
ИНТЕРПОЛИРОВАНИЕ ФУНКЦИИ С ПОМОЩЬЮ КУБИЧЕСКОГО СПЛАЙНА
***** }

```

```
program SplineInterpolation;
```

```
uses Crt;
```

```
const
```

```
    p = 15;
```

```
type
```

```
    vector=array[0..p] of real;
```

```
    parameter=array[0..p-1] of real;
```

```
    rightpart=array[1..p-1] of real;
```

```
var
```

```
    i,n:integer;
```

```
    a,b1,h,s,x,x0,x1:real;
```

```
    y,m:vector;
```

```
    b:rightpart;
```

```
    l:parameter;
```

```
    ch:char;
```

```
{ -----
                                ПОДПРОГРАММА                                }
```

```
procedure PAUSA;
```

```
    BEGIN
```

```
        WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
```

```
        REPEAT ch := readkey UNTIL ch <> '';
```

```
    END;
```

```
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                }
```

```
BEGIN
```

```
    ClrScr;
```

```
    WRITELN ('Введите n - число разбиений отрезка [a,b]');
```

```
    READ (n);
```

```
    WRITELN ('Введите координаты концов отрезка a, b');
```

```
    READ (a, b1);
```

```
    WRITELN ('Введите значения функции y(i) в узлах,');
```

```
    WRITELN (' причем y(0) = f(a), y(n) = f(b)');
```

```
    FOR i := 0 TO n DO READ (y[i]);
```

```
    WRITELN ('Введите значение x');
```

```
    READ (x);
```

```
{    Условия на концах отрезка: m(0)=f'(a)=1 и m(n)=f'(b)=0 }
```

```
    m[0] := 1; m[n] := 0; l[0] := 0; h := (b1 - a) / n;
```

```
    FOR i := 1 TO n - 1 DO
```

```

      BEGIN
        b[i] := 3 * (y[i + 1] - y[i - 1]) / h;
        l[i] := -1 / (l[i - 1] + 4);
        m[i] := l[i] * (m[i - 1] - b[i]);
      END;
    FOR i := n - 1 DOWNTO 1 DO
      m[i] := l[i] * m[i + 1] + m[i];
    IF (x > a) AND (x < b) THEN
      BEGIN
        i := TRUNC ((x - a) / h) + 1;
        x0 := a + (i - 1) * h;
        x1 := x0 + h;
        s := y[i - 1] * ((x - x1)*(x - x1) * (2 * (x - x0) +
          h)) / (h*h*h) + y[i] * ((x - x0)*(x - x0) * (2 * (x1 -
          x) + h)) / (h*h*h) + m[i - 1] * (x - x1)*(x - x1) * (x -
          x0) / (h*h) + m[i] * (x - x0)*(x - x0) * (x - x1) / (h*h);
        WRITE ('x = ', x, '    s = ', s);
      END;
    PAUSA;
  END.

```

4⁰. Программа построения кубического сплайна,
интерполирующего заданную функцию,
на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'

program SplineInterpolation
C      *****
C      ИНТЕРПОЛИРОВАНИЕ ФУНКЦИИ С ПОМОЩЬЮ КУБИЧЕСКОГО СПЛАЙНА
C      *****
      INTEGER*2 system
      REAL l(0:14),m(0:15)
      DIMENSION y(0:15),b(14)
      i=system('cls'C)
      WRITE (*,*) 'Введите n - число разбиений отрезка [a,b]'
      READ (*,*) n
      WRITE (*,*) 'Введите координаты концов отрезка a, b'
      READ (*,*) a, b
      WRITE (*,*) 'Введите n+1 значение функции y(i), '
      WRITE (*,*) ' причем y(0) = f(a), y(n) =f(b)'

```

```

DO i = 0, n
  READ (*, *) y(i)
END DO
WRITE (*, *) 'Введите значение x'
READ (*, *) x
C  Условия на концах отрезка: m(0)=f'(a)=1 и m(n)=f'(b)=0
m(0) = 1
m(n) = 0
l(0) = 0
h = (b1 - a) / n
DO i = 1, n - 1
  b(i) = 3 * (y(i + 1) - y(i - 1)) / h
  l(i) = -1 / (l(i - 1) + 4)
  m(i) = l(i) * (m(i - 1) - b(i))
END DO
DO i = n - 1, 1, -1
  m(i) = l(i) * m(i + 1) + m(i)
END DO
IF ((x.GT.a).AND.(x.LT.b1)) THEN
  i = INT((x - a) / h) + 1
  x0 = a + (i - 1) * h
  x1 = x0 + h
  s = y(i - 1) * ((x - x1)**2 * (2 * (x - x0) +
*   h)) / h**3 + y(i) * ((x - x0)**2 * (2 * (x1 -
*   x) + h)) / h**3 + m(i - 1) * (x - x1)**2 * (x -
*   x0) / h **2 + m(i) * (x - x0)**2 * (x - x1) / h**2
  WRITE (*, '(A5,E13.7,A7,E13.7)') ' x = ', x, '    s = ', s
END IF
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

5⁰. Программа построения кубического сплайна,
интерполирующего заданную функцию,
на языке C

```

/* *****
ИНТЕРПОЛИРОВАНИЕ ФУНКЦИИ С ПОМОЩЬЮ КУБИЧЕСКОГО СПЛАЙНА
***** */
#include <stdio.h>
#include <conio.h>

```



```

#include <math.h>
#define P 15
/* -----
                                ОСНОВНАЯ ПРОГРАММА                                */

void main ()
{
    int i,n;
    float a,b1,h,s,x,x0,x1,m[P+1],y[P+1],b[P],l[P];
    clrscr();
    printf ("Введите n - число разбиений отрезка [a,b]\n");
    scanf ("%i",&n);
    printf ("Введите координаты концов отрезка a, b\n");
    scanf ("%f%f",&a, &b1);
    printf ("Введите значения функции у(і) в узлах,");
    printf (" причем у(0) = f(a), у(n) = f(b)\n");
    for (i = 0; i <= n; scanf ("%f",&y[i]),i++);
    printf ("Введите значение х\n");
    scanf ("%f",&x);

/* Условия на концах отрезка: m(0)=f'(a)=1 и m(n)=f'(b)=0 */
    for (m[0]=1,m[n]=l[0]=0,h=(b1-a)/n,i=1; i <= n - 1; i++)
        { b[i] = 3 * (y[i + 1] - y[i - 1]) / h;
          l[i] = -1 / (l[i - 1] + 4);
          m[i] = l[i] * (m[i - 1] - b[i]);
        };
    for (i = n - 1; i >= 1; i--)
        m[i] = l[i] * m[i + 1] + m[i];

    if ((x > a) && (x < b1))
    { i = floor((x - a)/h) + 1 ;
      x0 = a + (i - 1) * h;
      x1 = x0 + h;
      s = y[i - 1] * ((x - x1)*(x - x1) * (2 * (x - x0) +
        h)) / (h*h*h) + y[i] * ((x - x0)*(x - x0) * (2 * (x1 -
        x) + h)) / (h*h*h) + m[i - 1] * (x - x1)*(x - x1) * (x -
        x0) / (h*h) + m[i] * (x - x0)*(x - x0) * (x - x1) / (h*h);
      printf ("x = %f   s = %f",x,s);
    };

    printf("\nНажмите любую клавишу для продолжения...");
    getch();
}

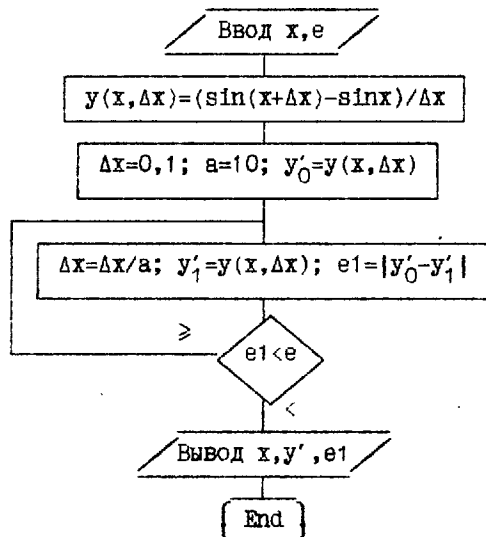
```

1⁰. Блок-схема вычисления производной по ее определению

При вычислении производной по ее определению находят отношение

$$\frac{\Delta y}{\Delta x} = \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

в некоторой точке x при уменьшении величины Δx до тех пор, пока разность этих отношений по абсолютной величине между двумя последовательными шагами вычислений не станет меньше, чем некоторое число ε . Тогда последний результат принимают в качестве производной функции, вычисленной в точке x с заданной точностью ε . В блок-схеме и программах вычислений рассматривается функция $y = \sin x$ (см. пример из §1).



2⁰. Программа вычисления производной по ее определению
на языке BASIC

```

1 REM *****
2 REM  ВЫЧИСЛЕНИЕ ПРОИЗВОДНОЙ ФУНКЦИИ ПО ЕЕ ОПРЕДЕЛЕНИЮ
3 REM    на примере функции                y = sin x
4 REM  *****
10 DEF fnv (x, d) = (SIN(x + d) - SIN(x)) / d
    
```

```

20 CLS
30 INPUT "Введите значение x = "; x
40 INPUT "Введите точность e = "; e
50 d = .1: a = 10: y1 = fny(x, d)
60 d = d / a: y2 = fny(x, d): e1 = ABS(y1 - y2)
70 IF e1 > e THEN y1 = y2: GOTO 60
80 a$ = "x = #.#### f'(x) = #.##### погрешность = #.#####"
90 PRINT USING a$; x; y2; e1
100 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
110 IF INKEY$ = "" THEN 110
120 END

```

3⁰. Программа вычисления производной по ее определению на языке PASCAL

```

program derivefunction;
{
  *****
  ВЫЧИСЛЕНИЕ ПРОИЗВОДНОЙ ФУНКЦИИ ПО ЕЕ ОПРЕДЕЛЕНИЮ
  на примере функции y = sinx
  *****
}
uses Crt;
const
  a=10;
var
  deltax, der1, der2, e, e1, x: real;
  ch: char;
{ -----
                                ПОДПРОГРАММЫ                                }
function derivf(x, deltax: real): real;
  BEGIN
    derivf := (sin(x+deltax) - sin(x))/deltax
  END;
{ ----- }
procedure Pausa;
  BEGIN
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
  END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                }

```

BEGIN

ClrScr;

WRITELN ('Введите значение x и точность epsilon');

READ (x,e);

deltax := 0.1; der1 := derivf(x,deltax);

REPEAT

deltax := deltax / a; der2 := derivf(x,deltax);

e1 := ABS(der1 - der2); der1 := der2;

UNTIL e1 < e;

WRITELN ('x = ',x,' произв = ',der2);

WRITELN ('Погрешность приближения = ',e1);

PAUSA;

END.

4⁰. Программа вычисления производной по ее определению на языке FORTRAN

\$INCLUDE: 'EXEC.FI'

program derivefunction

C *****

C ВЫЧИСЛЕНИЕ ПРОИЗВОДНОЙ ФУНКЦИИ ПО ЕЕ ОПРЕДЕЛЕНИЮ

C на примере функции y = sinx

C *****

INTEGER*2 system

derivf(x,deltax) = (sin(x+deltax)-sin(x))/deltax

i=system('cls'C)

WRITE (*,*) 'Введите значение x и точность epsilon'

READ (*,*) x,e

deltax = .1

a = 10

der1 = derivf(x,deltax)

e1 =1

DO WHILE (e1.GT.e)

deltax = deltax / a

der2 = derivf(x,deltax)

e1 = ABS(der1 - der2)

der1 = der2

END DO

```

WRITE (*, '(A4,E13.7,A9,E13.7)') ' x =', x, ' произв=', der2
WRITE (*, '(A26,E13.7)') ' Погрешность приближения =', e1
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

5⁰. Программа вычисления производной по ее определению на языке C

```

/* *****
   ВЫЧИСЛЕНИЕ ПРОИЗВОДНОЙ ФУНКЦИИ ПО ЕЕ ОПРЕДЕЛЕНИЮ
   НА ПРИМЕРЕ ФУНКЦИИ                               y = sinx
   ***** */
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define A 10
/* -----
                                ПОДПРОГРАММЫ                               */
float derivf(float x, float y)
{ float z; z = (sin(x+y) - sin(x))/y; return z; }
/* -----
                                ОСНОВНАЯ ПРОГРАММА                               */
void main()
{ float deltax, der1, der2, e, e1, x;
  clrscr();
  printf ("Введите значение x и точность epsilon\n");
  scanf ("%f%f", &x, &e);
  deltax = 0.1; der1 = derivf(x, deltax);
  do
  { deltax = deltax / A; der2 = derivf(x, deltax);
    e1 = fabs(der1 - der2); der1 = der2;
  }
  while (e1 >= e);
  printf ("\n\nx = %f произв f' = %f", x, der2);
  printf ("\nПогрешность приближения = %f\n", e1);
  printf ("\nНажмите любую клавишу для продолжения...");
  getch();
}

```

К §3

1⁰. Блок-схема вычисления производных
первого и второго порядков
с одинаковым порядком аппроксимации по шагу h

Пусть отрезок $[a, b]$ разбит на n равных частей с шагом $h = \frac{b-a}{n}$ и в точках x_i ($i=0, 1, 2, \dots, n$; $x_0=a$, $x_n=b$) некоторая функция принимает значения y_i . Для переменной x , принадлежащей отрезку $[a, b]$, требуется вычислить значения первой и второй производной, имеющих порядок аппроксимации $O(h^2)$ в зависимости от шага. Полагая, что значения производных y' и y'' в точках x , близких к точкам x_i , равны соответствующим значениям y'_i и y''_i . Будем считать точку x близкой к x_i , если она принадлежит промежутку $x_i - \frac{h}{2} \leq x \leq x_i + \frac{h}{2}$. Точки x , близкие к x_i , имеют одно и то же значение параметра

$$i = \text{целая часть} \left[\frac{x-a}{h} + \frac{1}{2} \right].$$

В зависимости от i при $n \geq 3$ имеем три типа формул (19) - (21) гл. 6. Так, при $i = 0$:

$$y'_0 \approx \frac{1}{2h} (-3y_0 + 4y_1 - y_2), \quad y''_0 \approx \frac{1}{h^2} (2y_0 - 5y_1 + 4y_2 - y_3);$$

при $i = 1, 2, \dots, n-1$:

$$y'_i \approx \frac{1}{2h} (-y_{i-1} + y_{i+1}), \quad y''_i \approx \frac{1}{h^2} (y_{i-1} - 2y_i + y_{i+1});$$

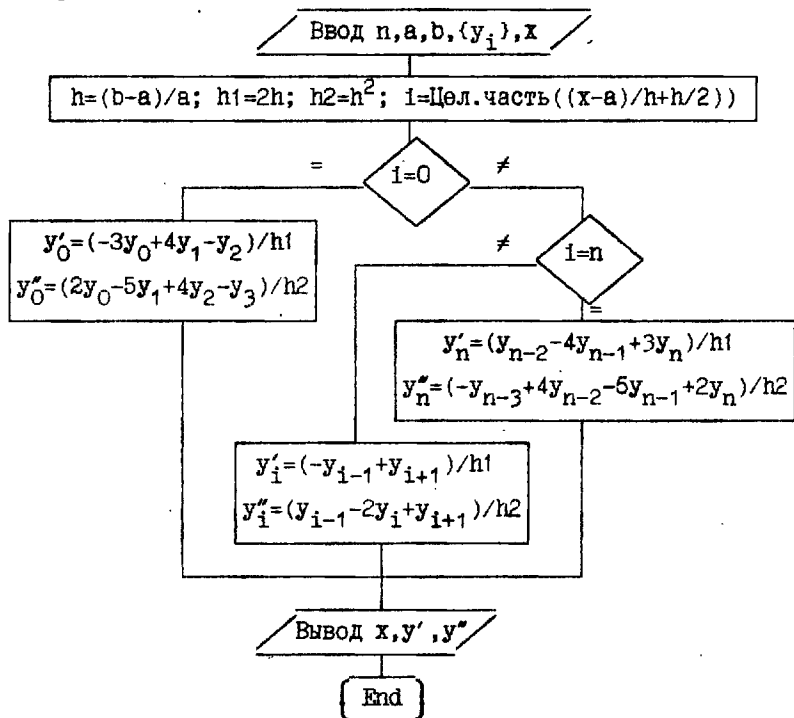
при $i = n$:

$$y'_n \approx \frac{1}{2h} (y_{n-2} - 4y_{n-1} + 3y_n), \quad y''_n \approx \frac{1}{h^2} (-y_{n-3} + 4y_{n-2} - 5y_{n-1} + 2y_n).$$

Чтобы получить правильные результаты, вычисления производят с двойной точностью. Для ориентировки в работе программ можно использовать таблицу

x	1	1,01	1,02	1,03
y'	2,718150	2,745650	2,773250	2,800950
y''	2,730000	2,750000	2,770000	2,790000
e^x	2,718282	2,745601	2,773195	2,801066

из которой, в частности, следует, что при вводе данных $n=3$; $a=1$; $b=1,03$ и четырех значений функции e^x с шестью верными знаками после запятой в результате вычислений сохраняется пять знаков после запятой для первой производной и два знака - для второй производной. Если, например, ввести значения функции с четырьмя знаками после запятой, то получим для y' значение 2,715000, а для y'' - 3,000000.



2⁰. Программа вычисления
производных первого и второго порядков
на языке BASIC

```

1 REM *****
2 REM  ВЫЧИСЛЕНИЕ ПРОИЗВОДНЫХ ПЕРВОГО И ВТОРОГО ПОРЯДКОВ
3 REM  С ОДИНАКОВОЙ ПОГРЕШНОСТЬЮ В ЗАВИСИМОСТИ ОТ ШАГА
4 REM  ПО ФОРМУЛАМ ЧИСЛЕННОГО ДИФФЕРЕНЦИРОВАНИЯ
5 REM *****
10 DEFINT I, N: DEFDBL A-B, H, X-Y: DIM y(15)
  
```

```

20 CLS
30 INPUT "Введите n >=3 - число разбиений отрезка [a,b] "; n
40 IF n < 3 OR n > 15 THEN 200
50 PRINT "Введите через запятую координаты концов отрезка a, b"
60 INPUT a, b
70 PRINT "Введите значения функции y(i) в узлах интерполяции"
80 PRINT "(i = 0, 1, 2, ..., n), причем y(0) = f(a), y(n) = f(b)"
90 FOR i = 0 TO n
100 INPUT y(i)
110 NEXT i
120 INPUT "Введите значение x "; x
130 h = (b - a) / n: i = INT((x - a) / h + h / 2)
140 h1 = 2 * h: h2 = h ^ 2
150 IF i = 0 THEN GOSUB 230
160 IF i > 0 AND i < n THEN GOSUB 260
170 IF i = n THEN GOSUB 290
180 PRINT USING "x = #.### произв1 = ##.##### "; x; y1;
190 PRINT USING " произв2 = ##.#####"; y2
200 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
210 IF INKEY$ = "" THEN 210
220 END
230 y1 = (-3 * y(0) + 4 * y(1) - y(2)) / h1
240 y2 = (2 * y(0) - 5 * y(1) + 4 * y(2) - y(3)) / h2
250 RETURN
260 y1 = (-y(i - 1) + y(i + 1)) / h1
270 y2 = (y(i - 1) - 2 * y(i) + y(i + 1)) / h2
280 RETURN
290 y1 = (y(n - 2) - 4 * y(n - 1) + 3 * y(n)) / h1
300 y2 = (-y(n - 3) + 4 * y(n - 2) - 5 * y(n - 1) + 2 * y(n)) / h2
310 RETURN

```

3⁰. Программа вычисления
производных первого и второго порядков
на языке PASCAL

```

{
*****
ВЫЧИСЛЕНИЕ ПРОИЗВОДНЫХ ПЕРВОГО И ВТОРОГО ПОРЯДКОВ
С ОДИНАКОВОЙ ПОГРЕШНОСТЬЮ В ЗАВИСИМОСТИ ОТ ШАГА
ПО ФОРМУЛАМ ЧИСЛЕННОГО ДИФФЕРЕНЦИРОВАНИЯ
*****
}

```



```

program DerivativeOf1and2order;
uses Grt;
const
    p = 15;
type
    vector=array[0..p] of real;
var
    i,n:integer;
    a,b,h,h1,h2,x,y1,y2:real;
    y:vector;
    ch:char;
{ -----
                                ПОДПРОГРАММА                                }
procedure PAUSA;
begin
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
    END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                }
begin
    ClrScr;
    WRITELN ('Введите n - число разбиений отрезка [a,b]');
    READ (n);
    IF (n >= 3) AND (n <= 15) THEN
        BEGIN
            WRITELN ('Введите координаты концов отрезка a, b');
            READ (a, b);
            WRITELN ('Введите значения функции y[i] в узлах,');
            WRITELN (' причем y(0) = f(a), y(n) =f(b)');
            FOR i := 0 TO n DO READ (y[i]);
            WRITELN ('Введите значение x');
            READ (x);
            h := (b - a) / n; i := TRUNC((x - a) / h + h/2);
            h1 := 2 * h; h2 := h * h;
            IF i = 0 THEN
                BEGIN
                    y1 := (-3 * y[0] + 4 * y[1] - y[2]) / h1;

```

```

      y2 := (2 * y[0] - 5 * y[1] + 4 * y[2] - y[3]) / h2;
    END;
  IF (1 > 0) AND (1 < n) THEN
    BEGIN
      y1 := (-y[1 - 1] + y[1 + 1]) / h1;
      y2 := (y[1 - 1] - 2 * y[1] + y[1 + 1]) / h2;
    END;
  IF 1 = n THEN
    BEGIN
      y1 := (y[n - 2] - 4 * y[n - 1] + 3 * y[n]) / h1;
      y2 := (-y[n - 3] + 4 * y[n - 2] - 5 * y[n - 1] + 2 * y[n]) / h2;
    END;
  WRITE ('x = ',x:6:3,' произв1 = ',y1:9:6);
  WRITELN (' произв2 = ',y2:9:6);
END;
PAUSA;
END.

```

4⁰. Программа вычисления
производных первого и второго порядков
на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
      program DerivativeOf1and2order
C      *****
C      ВЫЧИСЛЕНИЕ ПРОИЗВОДНЫХ ПЕРВОГО И ВТОРОГО ПОРЯДКОВ
C      С ОДИНАКОВОЙ ПОГРЕШНОСТЬЮ В ЗАВИСИМОСТИ ОТ ШАГА
C      ПО ФОРМУЛАМ ЧИСЛЕННОГО ДИФФЕРЕНЦИРОВАНИЯ
C      *****
      INTEGER*2 system
      DOUBLE PRECISION a,b,h,x,y
      DIMENSION y(0:15)
      i=system('cls'C)
      WRITE (*,*) 'Введите n - число разбиений отрезка [a,b]'
      READ (*,*) n
      IF ((n.GE.3).AND.(n.LE.15)) THEN
        WRITE (*,*) 'Введите координаты концов отрезка a, b'
        READ (*,*) a, b
        WRITE (*,*) 'Введите n+1 значение функции y(i), '

```

```

WRITE (*,*) ' причем  $y(0) = f(a)$ ,  $y(n) = f(b)$ '
DO 1 = 0,n
    READ (*,*) y(1)
END DO
WRITE (*,*) 'Введите значение x'
READ (*,*) x
h = (b - a) / n
1 = INT((x - a) / h + h / 2)
h1 = 2 * h
h2 = h * h
IF (1.EQ.0) THEN
    y1 = (-3 * y(0) + 4 * y(1) - y(2)) / h1
    y2 = (2 * y(0) - 5*y(1) + 4*y(2) - y(3)) / h2
END IF
IF ((1.GT.0).AND.(1.LT.n)) THEN
    y1 = (-y(1 - 1) + y(1 + 1)) / h1
    y2 = (y(1 - 1) - 2 * y(1) + y(1 + 1)) / h2
END IF
IF (1.EQ.n) THEN
    y1 = (y(n - 2) - 4 * y(n - 1) + 3 * y(n)) / h1
    y2 = (-y(n-3) + 4*y(n-2) - 5*y(n-1) + 2*y(n)) / h2
END IF
WRITE (*, '(A5,E10.4,A8,E12.6\)' ) ' x = ',x,' пр1 = ',y1
WRITE (*, '(A8,E12.6)' ) ' пр2 = ',y2
END IF
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

**5⁰. Программа вычисления
 производных первого и второго порядков
 на языке C**

```

/*
*****
ВЫЧИСЛЕНИЕ ПРОИЗВОДНЫХ ПЕРВОГО И ВТОРОГО ПОРЯДКОВ
С ОДИНАКОВОЙ ПОГРЕШНОСТЬЮ В ЗАВИСИМОСТИ ОТ ШАГА
ПО ФОРМУЛАМ ЧИСЛЕННОГО ДИФФЕРЕНЦИРОВАНИЯ
*****/
#include <stdio.h>
#include <conio.h>

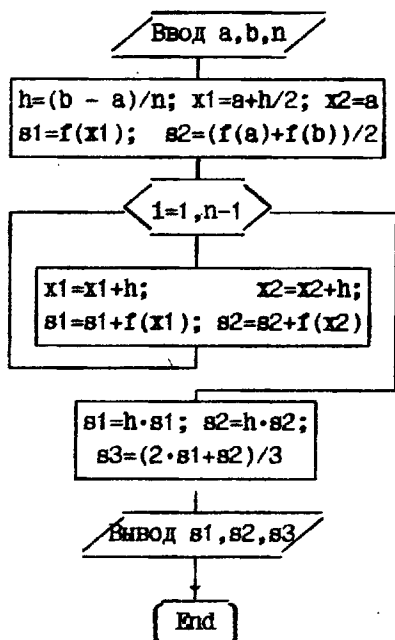
```

```

#include <math.h>
#define P 15
/* -----
                                ОСНОВНАЯ ПРОГРАММА                                */
void main()
{   int i,n;
    double a,b,h,h1,h2,x,y1,y2,y[P+1];
    clrscr();
    printf ("Введите n - число разбиений отрезка [a,b]\n");
    scanf ("%i",&n);
    if ((n >= 3) && (n <= 15))
    {   printf ("Введите координаты концов отрезка a, b\n");
        scanf ("%lf%lf",&a, &b);
        printf ("Введите значения функции y(i) в узлах,");
        printf (" причем y(0) = f(a), y(n) = f(b)\n");
        for (i = 0; i <= n; scanf ("%lf",&y[i]),i++);
        printf ("Введите значение x\n");
        scanf ("%lf",&x);
        h = (b - a) / n; i = floor((x - a) / h + h/2);
        h1 = 2 * h; h2 = h * h;
        if (i == 0)
        {   y1 = (-3 * y[0] + 4 * y[1] - y[2]) / h1;
            y2 = (2 * y[0] - 5 * y[1] + 4 * y[2] - y[3]) / h2;
        };
        if ((i > 0) && (i < n))
        {   y1 = (-y[i - 1] + y[i + 1]) / h1;
            y2 = (y[i - 1] - 2 * y[i] + y[i + 1]) / h2;
        };
        if (i == n)
        {   y1 = (y[n - 2] - 4 * y[n - 1] + 3 * y[n]) / h1;
            y2 = (-y[n - 3] + 4 * y[n - 2] - 5 * y[n - 1] + 2 * y[n]) / h2;
        };
        printf ("x = %f произв1 = %f",x,y1);
        printf (" произв2 = %f",y2);
    };
    printf ("\nНажмите любую клавишу для продолжения...");
    getch();
}

```

1⁰. Блок-схема вычисления определенного интеграла по квадратурным формулам прямоугольников, трапеций и Симпсона



a, b - концы отрезка интегрирования $[a, b]$;
 $f(x)$ - подынтегральная функция;
 n - число делений отрезка интегрирования

Приближенные значения интегралов по формулам
 прямоугольников - $s1$;
 трапеций - $s2$;
 Симпсона - $s3$

В программах к данному и следующему параграфам в качестве подынтегральной функции взята функция $f(x) = e^{x^2}$.

2⁰. Программа вычисления определенного интеграла по квадратурным формулам прямоугольников, трапеций и Симпсона

на языке BASIC

```

1 REM *****
2 REM ВЫЧИСЛЕНИЕ ОПРЕДЕЛЕННОГО ИНТЕГРАЛА ПО ФОРМУЛАМ
3 REM ПРЯМОУГОЛЬНИКОВ, ТРАПЕЦИИ И СИМПСОНА
4 REM *****
10 DEFINT I, N: CLS
20 PRINT "Введите через запятую значения концов отрезка [a,b]"
30 INPUT a, b
  
```

```

40 PRINT "Введите число разбиений отрезка n"
50 INPUT n
51 REM ----- ПОДЫНТЕГРАЛЬНАЯ ФУНКЦИЯ -----
60 DEF fnf (x) = EXP(x * x)
61 REM -----
70 h = (b - a) / n: x1 = a + h / 2: x2 = a
80 s1 = fnf(x1): s2 = (fnf(a) + fnf(b)) / 2
90 FOR i = 1 TO n - 1
100 x1 = x1 + h: x2 = x2 + h
110 s1 = s1 + fnf(x1): s2 = s2 + fnf(x2)
120 NEXT i
130 s1 = h * s1: s2 = h * s2: s3 = (2 * s1 + s2) / 3
140 PRINT "s1 = "; s1; " s2 = "; s2; " s3 = "; s3
150 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
160 IF INKEY$ = "" THEN 160
170 END

```

3⁰. Программа вычисления определенного интеграла
по квадратурным формулам прямоугольников, трапеций и Симпсона
на языке PASCAL

program integration;

```

{*****
  ВЫЧИСЛЕНИЕ ОПРЕДЕЛЕННОГО ИНТЕГРАЛА ПО ФОРМУЛАМ
    ПРЯМОУГОЛЬНИКОВ, ТРАПЕЦИИ И СИМПСОНА
  *****}

```

uses Crt;

var

1,n:integer;

a,b,h,s1,s2,s3,x,x1,x2:real;

ch:char;

```

{ -----
                      ПОДПРОГРАММЫ
                ПОДЫНТЕГРАЛЬНАЯ ФУНКЦИЯ                      }

```

function f(x:real):real;

BEGIN

f := exp(x*x)

END;

```

{ ----- }

```

```
procedure Pausa;
```

```
  BEGIN
```

```
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
```

```
    REPEAT ch := readkey UNTIL ch <> ' ';
```

```
  END;
```

```
( -----
                                ОСНОВНАЯ ПРОГРАММА                                )
```

```
BEGIN
```

```
  ClrScr;
```

```
  WRITELN ('Введите значения концов отрезка [a,b]');
```

```
  READ (a,b);
```

```
  WRITELN ('Введите число разбиений отрезка n');
```

```
  READ (n);
```

```
  h := (b - a) / n; x1 := a + h / 2; x2 := a;
```

```
  s1 := f(x1); s2 := (f(x2) + f(b)) / 2;
```

```
  FOR i := 1 TO n - 1 DO
```

```
    BEGIN
```

```
      x1 := x1 + h; x2 := x2 + h;
```

```
      s1 := s1 + f(x1); s2 := s2 + f(x2);
```

```
    END;
```

```
  s1 := h * s1; s2 := h * s2; s3 := (2 * s1 + s2) / 3;
```

```
  WRITELN ('s1 = ',s1:9:6,' s2 = ',s2:9:6,' s3 = ',s3:9:6);
```

```
  PAUSA;
```

```
END.
```

4⁰. Программа вычисления определенного интеграла
по квадратурным формулам прямоугольников, трапеций и Симпсона
на языке FORTRAN

```
$INCLUDE: 'EXEC.FI'
```

```
  program integration
```

```
C *****
```

```
C   ВЫЧИСЛЕНИЕ ОПРЕДЕЛЕННОГО ИНТЕГРАЛА ПО ФОРМУЛАМ
```

```
C   ПРЯМОУГОЛЬНИКОВ, ТРАПЕЦИЙ И СИМПСОНА
```

```
C *****
```

```
  INTEGER*2 system
```

```
C ----- ПОДЫНТЕГРАЛЬНАЯ ФУНКЦИЯ -----
```

```
  f(x) = exp(x**2)
```

```
C -----
```

```

i=system('cls'C)
WRITE (*,*) 'Введите значения концов отрезка [a,b]'
READ (*,*) a,b
WRITE (*,*) 'Введите число разбиений отрезка n'
READ (*,*) n
h = (b - a) / n
x1 = a + h / 2
x2 = a
s1 = f(x1)
s2 = (f(x2) + f(b)) / 2
DO i = 1, n - 1
    x1 = x1 + h
    x2 = x2 + h
    s1 = s1 + f(x1)
    s2 = s2 + f(x2)
END DO
s1 = h * s1
s2 = h * s2
s3 = (2 * s1 + s2) / 3
WRITE (*,'(3(A4,E13.7))') ' s1=',s1,' s2=',s2,' s3=',s3
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

5⁰. Программа вычисления определенного интеграла по квадратурным формулам прямоугольников, трапеций и Симпсона на языке C

```

/*****
  ВЫЧИСЛЕНИЕ ОПРЕДЕЛЕННОГО ИНТЕГРАЛА ПО ФОРМУЛАМ
    ПРЯМОУГОЛЬНИКОВ, ТРАПЕЦИИ И СИМПСОНА
  *****/
#include <stdio.h>
#include <conio.h>
#include <math.h>

/* -----
                                ПОДПРОГРАММА
                                Подынтегральная функция
float f(float x)
                                */

```



```

( float y; y = exp(x*x); return y; );
/* -----
                                ОСНОВНАЯ ПРОГРАММА                                */

void main()
( int i,n;
  float a,b,h,s1,s2,s3,x,x1,x2;
  clrscr();
  printf ("Введите значения концов отрезка [a,b]\n");
  scanf ("%f%f",&a,&b);
  printf ("Введите число разбиений отрезка n\n");
  scanf ("%i",&n);
  h = (b - a) / n; x1 = a + h / 2; x2 = a;
  s1 = f(x1); s2 = (f(x2) + f(b)) / 2;
  for (i = 1; i <= n - 1; i++)
    { x1 += h; x2 += h; s1 += f(x1); s2 += f(x2);
      };
  s1 *= h; s2 *= h; s3 = (2 * s1 + s2) / 3;
  printf ("\ns1 = %f s2 = %f s3 = %f",s1,s2,s3);
  printf("\n\nНажмите любую клавишу для продолжения...");
  getch();
}

```

1¹. Блок-схема вычисления определенного интеграла методом двойного пересчета по формуле Симпсона

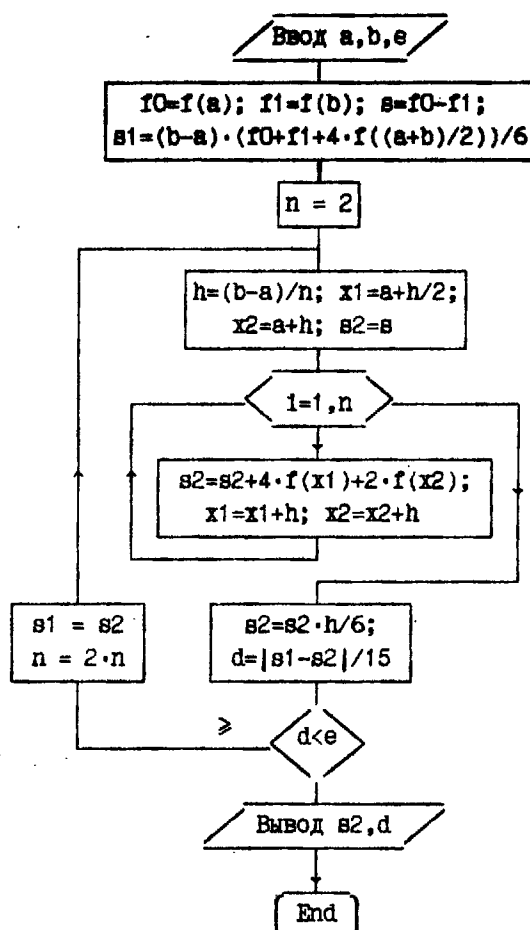
Вычисления интеграла по формуле Симпсона

$$Z = \int_a^b f(x) dx \approx \frac{h}{6} \left(y_0 - y_n + \sum_{i=1}^n (4(y_{i-1/2}) + 2y_i) \right)$$

ведут с увеличением числа разбиений n вдвое до окончания приближений при выполнении условия

$$\frac{1}{15} |Z(n) - Z(2n)| < \varepsilon.$$

При этом полагают $Z \approx Z(2n)$ с точностью ε .



2¹. Программа вычисления определенного интеграла
методом двойного пересчета по формуле Симпсона
на языке BASIC

```

1 REM *****
2 REM ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ epsilon
3 REM МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ СИМПСОНА
4 REM *****
10 DEFINT I, N: CLS
20 PRINT "Введите через запятую значения концов отрезка [a,b]"
30 INPUT a, b

```

```

40 PRINT "Введите точность вычисления  epsilon"
50 INPUT e
60 DEF fnf (x) = EXP(x * x)
70 f0 = fnf(a): f1 = fnf(b): s = f0 - f1
80 s1 = (b - a) * (f0 + f1 + 4 * fnf((a + b) / 2)) / 6
90 n = 2
100 h = (b - a) / n: x1 = a + h / 2: x2 = a + h: s2 = s
110 FOR i = 1 TO n
120     s2 = s2 + 4 * fnf(x1) + 2 * fnf(x2)
130     x1 = x1 + h: x2 = x2 + h
140 NEXT i
150 s2 = s2 * h / 6: d = ABS(s1 - s2) / 15
160     IF d < e THEN GOTO 180
170     s1 = s2: n = 2 * n: GOTO 100
180 PRINT "Величина интеграла s ="; s2; " Погрешность d = "; d
190 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
200 IF INKEY$ = "" THEN 200
210 END

```

3¹. Программа вычисления определенного интеграла
методом двойного пересчета по формуле Симпсона
на языке PASCAL

```

program IntegrationBySimpson;
(*****
  ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ epsilon
  МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ СИМПСОНА
  *****)
uses Crt;
var
  i,n:integer;
  a,b,d,e,f0,f1,h,s,s1,s2,x,x1,x2:real;
  ch:char;
(-----
                                ПОДПРОГРАММЫ
                                ПОДИНТЕГРАЛЬНАЯ ФУНКЦИЯ                                )
function f(x:real):real;
BEGIN
  f := exp(x*x)
END;

```

```

( ----- )
procedure Pausa;
  BEGIN
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
  END;
( ----- )

```

ОСНОВНАЯ ПРОГРАММА

```

BEGIN
  ClrScr;
  WRITELN ('Введите значения концов отрезка [a,b]');
  READ (a,b);
  WRITELN ('Введите точность вычисления epsilon');
  READ (e);
  f0 := f(a); f1 := f(b); s := f0 - f1;
  s1 := (b - a) * (f0 + f1 + 4 * f((a + b) / 2)) / 6;
  n := 2;
  REPEAT
    h := (b - a) / n; x1 := a + h / 2; x2 := a + h; s2 := s;
    FOR i := 1 TO n DO
      BEGIN
        s2 := s2 + 4 * f(x1) + 2 * f(x2);
        x1 := x1 + h; x2 := x2 + h;
      END;
    s2 := s2 * h / 6; d := ABS(s1 - s2) / 15;
    s1 := s2; n := 2 * n;
  UNTIL d < e;
  WRITELN ('Величина интеграла s =', s2:9:6, ' Погрешн. d =', d:9:6);
  PAUSA;
END.

```

4¹. Программа вычисления определенного интеграла
методом двойного пересчета по формуле Симпсона
на языке FORTRAN

```
$INCLUDE: 'EXEC.PI'
```

```
  program IntegrationBySimpson
```

```

C *****
C ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ epsilon
C МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ СИМПСОНА

```

```

C      *****
C      INTEGER*2 system
C      _____ ПОДЫНТЕГРАЛЬНАЯ ФУНКЦИЯ _____
C      f(x) = exp(x**2)
C      _____
      i=system('cls'C)
      WRITE (*,*) 'Введите значения концов отрезка [a,b]'
      READ (*,*) a,b
      WRITE (*,*) 'Введите точность вычисления epsilon'
      READ (*,*) e
      f0 = f(a)
      f1 = f(b)
      s1 = (b - a) * (f0 + f1 + 4 * f((a + b) / 2)) / 6
      s = f0 - f1
      n = 2
      d = 1
      DO WHILE (d.GT.e)
         h = (b - a) / n
         x1 = a + h / 2
         x2 = a + h
         s2 = s
         DO i = 1,n
            s2 = s2 + 4 * f(x1) + 2 * f(x2)
            x1 = x1 + h
            x2 = x2 + h
         END DO
         s2 = s2 * h / 6
         d = ABS(s1 - s2)/15
         s1 = s2
         n = 2 * n
      END DO
      WRITE (*, '(2(A12,E13.7))') ' Вел. ин. s=',s2,' Погр. d=',d
      PAUSE 'Нажмите клавишу ENTER для продолжения...'
      END
      51. Программа вычисления определенного интеграла
            методом двойного пересчета по формуле Симпсона
            на языке C

```

/*****

ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ EPSILON МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ СИМПСОНА

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <math.h>
```

```
/* -----
```

ПОДПРОГРАММЫ

Подынтегральная функция */

```
float f(float x)
```

```
{ float y; y = exp(x*x); return y; }
```

```
/* -----
```

ОСНОВНАЯ ПРОГРАММА

*/

```
void main()
```

```
{ int i,n;
```

```
float a,b,d,e,f0,f1,h,s,s1,s2,x,x1,x2;
```

```
clrscr();
```

```
printf ("Введите значения концов отрезка [a,b]\n");
```

```
scanf ("%f%f",&a,&b);
```

```
printf ("Введите точность вычисления epsilon\n");
```

```
scanf ("%f",&e);
```

```
f0 = f(a); f1 = f(b); s = f0 - f1;
```

```
s1 = (b - a) * (f0 + f1 + 4 * f((a + b) / 2)) / 6;
```

```
n = 2;
```

```
do
```

```
{ h = (b - a) / n; x1 = a + h / 2; x2 = a + h; s2=s;
```

```
for (i = 1; i <= n; i++)
```

```
{ s2 = s2 + 4 * f(x1) + 2 * f(x2);
```

```
x1 += h; x2 += h;
```

```
};
```

```
s2 *= h / 6; d = fabs(s1 - s2)/15;
```

```
s1 = s2; n *= 2;
```

```
}
```

```
while (d >= e);
```

```
printf ("Велич. интегр. s = %f Погрешн. d = %f",s2,d);
```

```
printf ("\n\nНажмите любую клавишу для продолжения...");
```

```
getch();
```

```
}
```

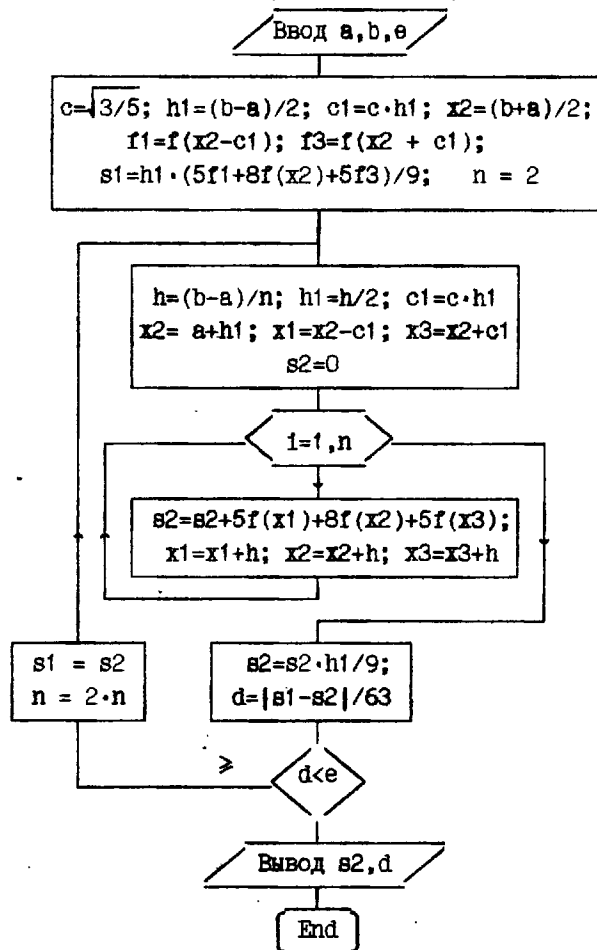
К §2

1⁰. Блок-схема вычисления определенного интеграла методом двойного пересчета по формуле Гаусса с тремя узлами

Вычисления интеграла по формуле Гаусса с тремя узлами

$$Z = \int_a^b f(x) dx \sim \frac{h}{18} \sum_{i=1}^n [5f(x_{i1}) + 8f(x_{i2}) + 5f(x_{i3})]$$

ведут с увеличением числа разбиений n вдвое до окончания приближений, исходя из условия $|Z(n) - Z(2n)|/63 < \varepsilon$.



**2⁰. Программа вычисления определенного интеграла
методом двойного пересчета по формуле Гаусса с тремя узлами
на языке BASIC**

```

1 REM *****
2 REM ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ epsilon
3 REM МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ ГАУССА
4 REM *****
10 DEFINT I, N: CLS
20 PRINT "Введите через запятую значения концов отрезка [a,b]"
30 INPUT a, b
40 PRINT "Введите точность вычисления epsilon"
50 INPUT e
51 REM -----
52 REM          ПОДЫНТЕГРАЛЬНАЯ ФУНКЦИЯ
60 DEF fnf (x) = EXP(x * x)
61 REM -----
70 c = SQR(3 / 5): h1 = (b - a) / 2: c1 = c * h1:
80 x2 = (b + a) / 2: f1 = fnf(x2 - c1): f3 = fnf(x2 + c1)
90 s1 = h1 * (5 * f1 + 8 * fnf(x2) + 5 * f3) / 9
100 n = 2
110 h = (b - a) / n: h1 = h / 2: c1 = c * h1
120 x2 = a + h1: x1 = x2 - c1: x3 = x2 + c1
130 s2 = 0
140 FOR i = 1 TO n
150 s2 = s2 + 5 * fnf(x1) + 8 * fnf(x2) + 5 * fnf(x3)
160 x1 = x1 + h: x2 = x2 + h: x3 = x3 + h
170 NEXT i
180 s2 = s2 * h1 / 9: d = ABS(s1 - s2) / 63
190 IF d < e THEN GOTO 210
200 s1 = s2: n = 2 * n: GOTO 110
210 PRINT "Величина интеграла s = "; s2; " Погрешность d = "; d
220 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
230 IF INKEY$ = "" THEN 230
240 END

```


**3⁰. Программа вычисления определенного интеграла
методом двойного пересчета по формуле Гаусса с тремя узлами
на языке PASCAL**

```

program IntegrationByGauss;
{*****
  ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ epsilon
  МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ ГАУССА
  *****)
uses Crt;
var
  i,n:integer;
  a,b,c,c1,d,e,f1,f3,h,h1,s,s1,s2,x1,x2,x3:real;
  ch:char;
{ -----
                                ПОДПРОГРАММЫ
                                ПОДЫНТЕГРАЛЬНАЯ ФУНКЦИЯ
                                }
function f(x:real):real;
  BEGIN
    f := exp(x*x)
  END;
{ -----
procedure Pausa;
  BEGIN
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
  END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА
                                }
BEGIN
  ClrScr;
  WRITELN ('Введите значения концов отрезка [a,b]');
  READ (a,b);
  WRITELN ('Введите точность вычисления epsilon');
  READ (e);
  c := SQRT(3 / 5); h1 := (b - a) / 2; c1 := c * h1;
  x2 := (b + a) / 2; f1 := f(x2 - c1); f3 := f(x2 + c1);
  s1 := h1 * (5 * f1 + 8 * f(x2) + 5 * f3) / 9;

```

```

n := 2;
REPEAT
  h := (b - a) / n; h1 := h / 2; c1 := c * h1;
  x2 := a + h1; x1 := x2 - c1; x3 := x2 + c1; s2 := 0;
  FOR i := 1 TO n DO
    BEGIN
      s2 := s2 + 5 * f(x1) + 8 * f(x2) + 5 * f(x3);
      x1 := x1 + h; x2 := x2 + h; x3 := x3 + h;
    END;
  s2 := s2 * h1 / 9; d := ABS(s1 - s2)/63;
  s1 := s2; n := 2*n;
UNTIL d < e;
WRITELN ('Величина интеграла s =',s2:9:6,' Погрешн. d=',d:9:6);
PAUSA;
END.

```

4⁰. Программа вычисления определенного интеграла
методом двойного пересчета по формуле Гаусса с тремя узлами
на языке FORTRAN

```

$INCLUDE: 'EXEC.PI'

program IntegrationByGauss
C *****
C ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ epsilon
C МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ ГАУССА
C *****
INTEGER*2 system
C ----- ПОДЫНТЕГРАЛЬНАЯ ФУНКЦИЯ -----
f(x) = exp(x**2)
C -----
i=system('cls')
WRITE (*,*) 'Введите значения концов отрезка [a,b]'
READ (*,*) a,b
WRITE (*,*) 'Введите точность вычисления epsilon'
READ (*,*) e
c = SQRT(3./5.)
h1 = (b - a) / 2
c1 = c * h1

```

```

x2 = (b + a) / 2
f1 = f(x2 - c1)
f3 = f(x2 + c1)
s1 = h1 * (5 * f1 + 8 * f(x2) + 5 * f3) / 9
n = 2
d = 1
DO WHILE (d.GT.e)
    h = (b - a) / n
    h1 = h / 2
    c1 = c * h1
    x2 = a + h1
    x1 = x2 - c1
    x3 = x2 + c1
    s2 = 0
    DO i = 1,n
        s2 = s2 + 5 * f(x1) + 8 * f(x2) + 5 * f(x3)
        x1 = x1 + h
        x2 = x2 + h
        x3 = x3 + h
    END DO
    s2 = s2 * h1 / 9
    d = ABS(s1 - s2)/63
    s1 = s2
    n = 2 * n
END DO
WRITE (*, '(2(A12,E13.7))') ' Вел. ин. s=', s2, ' Погр. d=', d
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

**5⁰. Программа вычисления определенного интеграла
методом двойного пересчета по формуле Гаусса с тремя узлами
на языке C**

```

/*****
ВЫЧИСЛЕНИЕ ИНТЕГРАЛА С ЗАДАННОЙ ТОЧНОСТЬЮ EPSILON
МЕТОДОМ ДВОЙНОГО ПЕРЕСЧЕТА ПО ФОРМУЛЕ ГАУССА
*****/
#include <stdio.h>

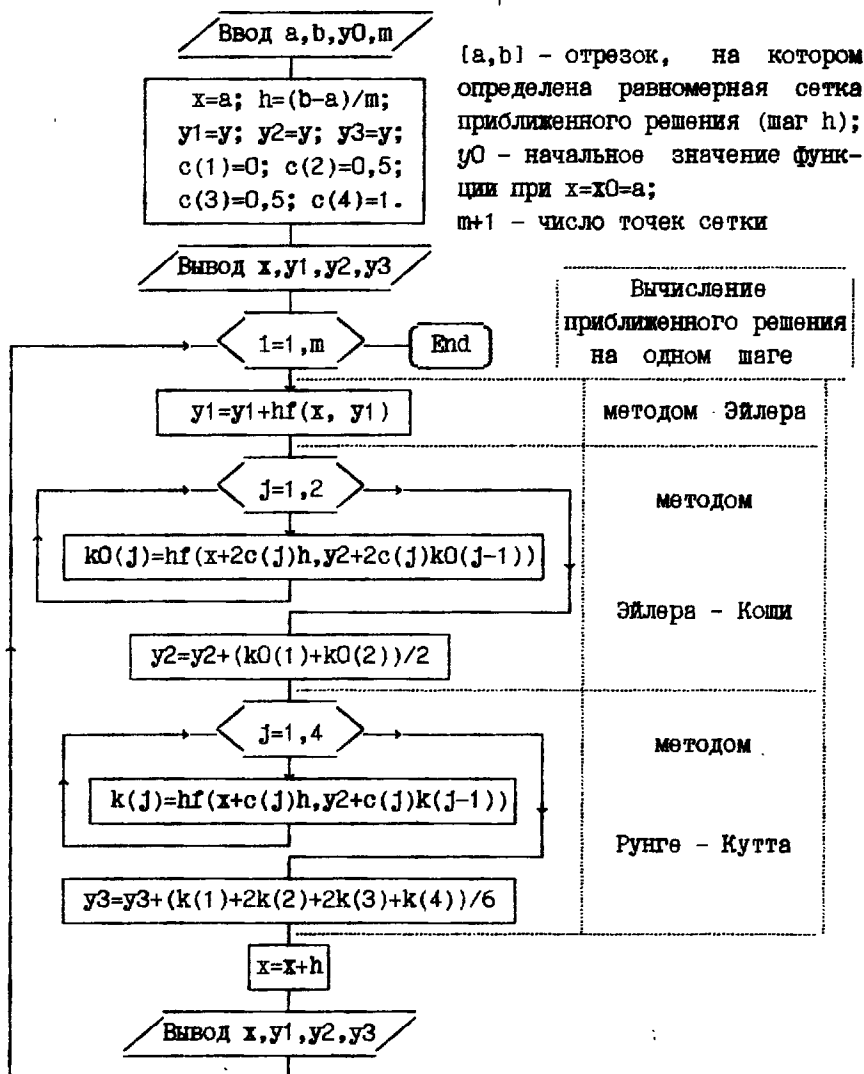
```

```

#include <conio.h>
#include <math.h>
#define C sqrt(3.0 / 5.0)
/* -----
                ПОДПРОГРАММА
                Подынтегральная функция                */
float f(float x)
{ float y; y = exp(x*x); return y; }
/* -----
                ОСНОВНАЯ ПРОГРАММА                */
void main()
{ int i,n;
  float a,b,c1,d,e,f1,f3,h,h1,s1,s2,x1,x2,x3;
  clrscr();
  printf ("Введите значения концов отрезка [a,b]\n");
  scanf ("%f%f",&a,&b);
  printf ("Введите точность вычисления epsilon\n");
  scanf ("%f",&e);
  h1 = (b - a) / 2; c1 = C * h1;
  x2 = (b + a) / 2; f1 = f(x2 - c1); f3 = f(x2 + c1);
  s1 = h1 * (5 * f1 + 8 * f(x2) + 5 * f3) / 9;
  n = 2;
do
{ h = (b - a) / n; h1 = h / 2; c1 = C * h1;
  x2 = a + h1; x1 = x2 - c1; x3 = x2 + c1; s2 = 0;
  for (i = 1; i <= n; i++)
  { s2 = s2 + 5 * f(x1) + 8 * f(x2) + 5 * f(x3);
    x1 += h; x2 += h; x3 += h;
  };
  s2 *= h1 / 9; d = fabs(s1 - s2)/63;
  s1 = s2; n *= 2;
}
while (d >= e);
printf ("Велич. интегр. = %f Погрешн. d = %f",s2,d);
printf("\n\nНажмите любую клавишу для продолжения...");
getch();
}

```

1⁰. Блок-схема численного решения задачи Коши для дифференциального уравнения первого порядка методами Эйлера, Эйлера - Коши и Рунге - Кутты



В программах к этому параграфу рассматривается задача Коши $y' = x+y$, $y|_{x=x_0} = y_0$. На экран выводятся приближенные решения на отрезке $[a,b]$ в виде таблицы $m+1$ значений функции.

**2⁰. Программа численного решения задачи Коши
для дифференциального уравнения первого порядка
методами Эйлера, Эйлера - Коши и Рунге - Кутты
на языке BASIC**

```

1  REM *****
2  REM    РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ ДИФФЕРЕНЦИАЛЬНОГО УРАВНЕНИЯ
3  REM    ПЕРВОГО ПОРЯДКА МЕТОДАМИ
4  REM    ЭЙЛЕРА, ЭЙЛЕРА - КОШИ, РУНГЕ - КУТТА
5  REM *****
6  REM Правая часть дифференциального уравнения - функция f(x,y)
10 DEFINT I-J, M: DIM c(4), k0(2), k(4)
20 DEF fnf (x, y) = x + y
30 CLS
40 PRINT "Введите через запятую значения концов отрезка [a,b]"
50 INPUT a, b
60 INPUT "Введите начальное значение функции y0 при x=x0 "; y
70 INPUT "Введите число шагов на отрезке [a,b]"; m
80    x = a: h = (b - a) / m: y1 = y: y2 = y: y3 = y
90 a$ = "x = ##.##  y1 = #.#####  y2 = #.#####  y3 = #.#####"
100 b$ = "          Мет. Эйлера    Мет. Э.-Коши    Мет. Рунге-К."
110 PRINT : PRINT b$
120 PRINT USING a$: x; y1; y2; y3
130    c(1) = 0: c(2) = .5: c(3) = .5: c(4) = 1
140 FOR i = 1 TO m
150    y1 = y1 + h * fnf(x, y1)
160    FOR j = 1 TO 2
170    k0(j) = h * fnf(x + 2 * c(j) * h, y2 + 2 * c(j) * k0(j - 1))
180    NEXT j
190    y2 = y2 + (k0(1) + k0(2)) / 2
200    FOR j = 1 TO 4
210    k(j) = h * fnf(x + c(j) * h, y3 + c(j) * k(j - 1))
220    NEXT j
230    y3 = y3 + (k(1) + 2 * k(2) + 2 * k(3) + k(4)) / 6

```

```

240   x = x + h
250   PRINT USING a$; x; y1; y2; y3
260 NEXT 1
270 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
280 IF INKEY$ = "" THEN 280
290 END

```

3⁰. Программа численного решения задачи Коши для дифференциального уравнения первого порядка методами Эйлера, Эйлера - Коши и Рунге - Кутты на языке PASCAL

```

program DifEquationsOfFirstOrder;
( *****
  РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ ДИФФЕРЕНЦИАЛЬНОГО УРАВНЕНИЯ
    ПЕРВОГО ПОРЯДКА МЕТОДАМИ
    ЭЙЛЕРА, ЭЙЛЕРА - КОШИ, РУНГЕ - КУТТА
  ***** )
uses Crt;
const c:array[1..4] of real =(0,0.5,0.5,1);
type
  coef= array[0..4] of real;
var
  i,j,m:integer;
  a,b,h,x,y,y1,y2,y3:real;
  k0,k:coef;
  ch:char;
( -----
                                ПОДПРОГРАММЫ                                )
( Правая часть дифференциального уравнения - функция f(x,y) )
function f(x,y:real):real;
  BEGIN
    f := x + y
  END;
( ----- )
procedure Pausa;
  BEGIN
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
  END;

```

```

{ -----
                                ОСНОВНАЯ ПРОГРАММА                                }
BEGIN
  ClrScr;
  WRITELN ('Введите значения концов отрезка [a,b]');
  READ (a, b);
  WRITELN ('Введите начальное значение функции у0 при x=x0');
  READ (y);
  WRITELN ('Введите число значений функции на промежутке (a,b]');
  READ (m);
  x := a; h := (b - a) / m; y1 := y; y2 := y; y3 := y;
  WRITELN ('          Метод Эйлера Метод Э.-Коши Метод Рунге-К.');
  WRITELN ('x=',x:5:2,' y1=',y1:9:6,' y2=',y2:9:6,' y3=',y3:9:6);
  FOR i := 1 TO m DO
    BEGIN
      y1 := y1 + h * f(x, y1); { <-----! Метод Эйлера}
      FOR j := 1 TO 2 DO
        k0[j] := h*f(x+2*c[j]*h, y2+2*c[j]* k0[j-1]);
      y2 := y2 + (k0[1] + k0[2]) / 2; { <-----! Метод Эйлера-Коши}
      FOR j := 1 TO 4 DO
        { <-----! Метод Рунге-Кутты}
        k[j] := h * f(x + c[j] * h, y3 + c[j] * k[j - 1]);
        y3 := y3 + (k[1] + 2*k[2] + 2*k[3] + k[4]) / 6;
        x := x + h;
      WRITELN ('x=',x:5:2,' y1=',y1:9:6,' y2=',y2:9:6,' y3=',y3:9:6);
    END;
  PAUSA;
END.

```

4⁰. Программа численного решения задачи Коши
 для дифференциального уравнения первого порядка
 методами Эйлера, Эйлера - Коши и Рунге - Кутта
 на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
program DifEquationsOfFirstOrder
C *****
C      РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ ДИФФЕРЕНЦИАЛЬНОГО УРАВНЕНИЯ
C      ПЕРВОГО ПОРЯДКА МЕТОДАМИ
C      ЭЙЛЕРА, ЭЙЛЕРА - КОШИ, РУНГЕ - КУТТА

```



```

C *****
INTEGER*2 system
REAL k0(0:2),k(0:4)
DIMENSION c(4)
C Правая часть дифференциального уравнения - функция f(x,y)
f(x,y) = x + y
i=system('cls'C)
WRITE (*,*) 'Введите значения концов отрезка [a,b]'
READ (*,*) a,b
WRITE (*,*) 'Введите начальное значение функции y0 при x=x0'
READ (*,*) y
WRITE (*,*) 'Введите число шагов на отрезке [a,b]'
READ (*,*) m
  x = a
  h = (b - a) / m
  y1 = y
  y2 = y
  y3 = y
WRITE(*, '(4(A5,E13.7))') ' x=',x,' y1=',y1,' y2=',y2,' y3=',y3
  c(1) = 0
  c(2) = 0.5
  c(3) = 0.5
  c(4) = 1
  DO i = 1,m
    y1 = y1 + h * f(x, y1)
    DO j = 1, 2
      k0(j)=h*f(x+2*c(j)*h, y2+2*c(j)*k0(j - 1))
    END DO
    y2 = y2 + (k0(1) + k0(2)) / 2
    DO j = 1, 4
      k(j)=h*f(x + c(j) * h, y3 + c(j) * k(j - 1))
    END DO
    y3 = y3 + (k(1) + 2 * k(2) + 2 * k(3) + k(4)) / 6
    x = x + h
  WRITE(*, '(4(A5,E13.7))') ' x=',x,' y1=',y1,' y2=',y2,' y3=',y3
  END DO
  PAUSE 'Нажмите клавишу ENTER для продолжения...'
  END

```

**5⁰. Программа численного решения задачи Коши
для дифференциального уравнения первого порядка
методами Эйлера, Эйлера - Коши и Рунге - Кутты
на языке C**

```

/*****
РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ ДИФФЕРЕНЦИАЛЬНОГО УРАВНЕНИЯ
ПЕРВОГО ПОРЯДКА МЕТОДАМИ ЭЙЛЕРА, ЭЙЛЕРА - КОШИ, РУНГЕ - КУТТА
*****/
#include <stdio.h>
#include <conio.h>

/* ----- ПОДПРОГРАММА -----
Правая часть дифференциального уравнения - функция f(x,y)*/
float f(float x, float y)
( float z; z = x + y; return z; )

/* ----- ОСНОВНАЯ ПРОГРАММА ----- */
void main()
( int i,j,m;
  float a,b,h,x,y,y1,y2,y3,k0[5],k[5];
  float c[4]={0,.5,.5,1};
  clrscr();
  printf ("Введите значения концов отрезка [a,b]\n");
  scanf ("%f%f",&a,&b);
  printf ("Введите начальное значение функции у0 при x=x0\n");
  scanf ("%f",&y);
  printf ("Введите число значений функции на промежутке (a,b)\n");
  scanf ("%i",&m);
  x = a; h = (b - a) / m; y1 = y2 = y3 = y;
  printf ("          Метод Эйлера Метод Э.-Коши Метод Рунге-К.\n");
  printf ("x=%-5.2f y1=%f y2=%f y3=%f\n",x,y1,y2,y3);
  for (i = 1; i <= m; i++)
  {
    y1 += h * f(x, y1);          /* <-----! Метод Эйлера*/
    for (j = 1; j <= 2; j++)
      k0[j] = h*f(x+2*c[j-1]*h, y2+2*c[j-1]* k0[j-1]);
    y2 += (k0[1] + k0[2]) / 2;   /* <-----! Метод Эйлера-Коши*/
    for (j = 1; j <= 4; j++)    /* <-----! Метод Рунге-Кутта*/
      k[j] = h * f(x + c[j-1] * h, y3 + c[j-1] * k[j - 1]);
    y3 += (k[1] + 2*k[2] + 2*k[3] + k[4]) / 6;
  }

```

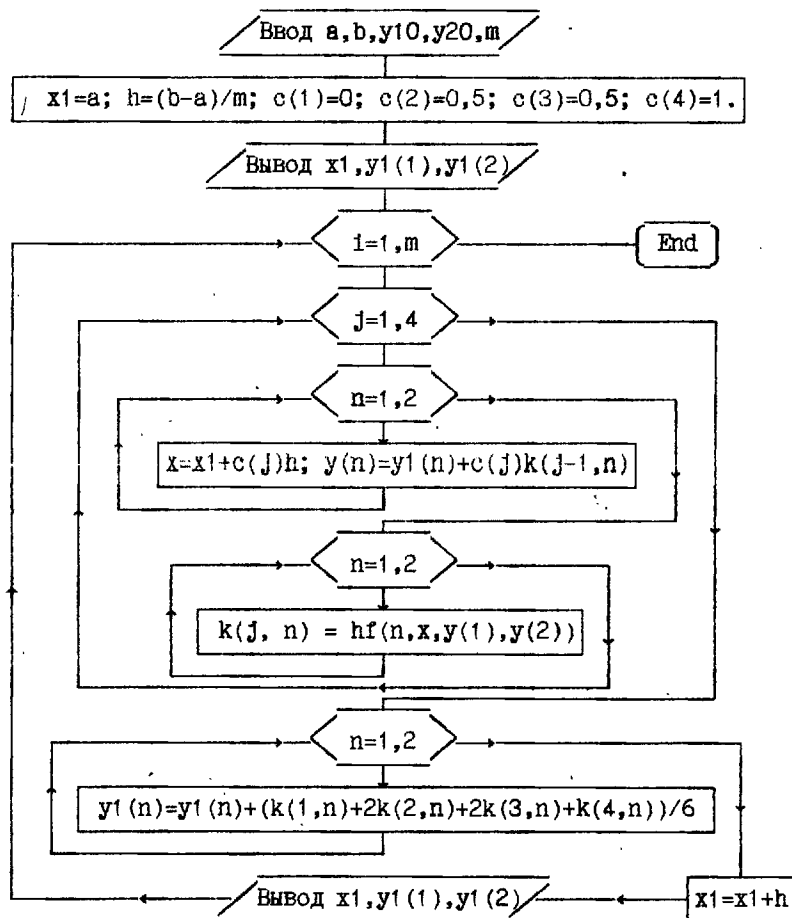
```

x += h;
printf ("x=%-5.2f y1=%f y2=%f y3=%f\n",x,y1,y2,y3);
);
printf("\n\nНажмите любую клавишу для продолжения...");
getch();
)

```

К §2

Блок-схема численного решения задачи Коши
для системы дифференциальных уравнений первого порядка
методом Рунге - Кутты



В программах к этому параграфу рассматривается задача Коши для системы дифференциальных уравнений первого порядка

$$\begin{cases} y_1' = y_2, \\ y_2' = -y_1; \end{cases}$$

$$y_1|_{x=x_0}=y_{10}, \quad y_2|_{x=x_0}=y_{20}.$$

Задача Коши для дифференциального уравнения второго порядка

$$y'' + y = 0, \quad y|_{x=x_0}=y_0, \quad y'|_{x=x_0}=y_0'$$

приводится к задаче Коши для предыдущей системы, если обозначить

$$y_1(x) = y(x), \quad y_2 = y_1'(x) \quad \text{и} \quad y_{10} = y_0, \quad y_{20} = y_0'.$$

Вычисления правых частей дифференциальных уравнений

$$f_1(x, y_1, y_2) = y_2,$$

$$f_2(x, y_1, y_2) = -y_1$$

ведутся в подпрограммах. Заметим, что явной зависимости от x в данных функциях нет и, в частности, при компиляции программы на языке FORTRAN на экране появляется предупреждение о неиспользованной переменной x .

При выполнении программы на экран выводятся приближенные решения на отрезке $[a, b]$ в виде таблицы $m+1$ значений двух функций на равномерной сетке с шагом h .

2⁰. Программа численного решения задачи Коши для системы дифференциальных уравнений первого порядка методом Рунге - Кутта на языке BASIC

```
1  REM *****
2  REM РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ СИСТЕМЫ ДИФФЕРЕНЦИАЛЬНЫХ
3  REM УРАВНЕНИЙ ПЕРВОГО ПОРЯДКА МЕТОДОМ РУНГЕ - КУТТА
4  REM *****
10 DEFINT I-J, M-N: DIM c(4), k(4, 2), y(2), y1(2)
20 CLS
30 PRINT "Введите через запятую значения концов отрезка [a,b]"
40 INPUT a, b
50 PRINT "Введите через запятую начальные значения"
60 PRINT "функции y10 и функции y20 при x=x0 ";
70 INPUT y1(1), y1(2)
```

```

80 INPUT "Введите число шагов на отрезке [a,b]"; m
90 a$ = "x = ##.## y1 = ##.##### y2 = ##.##### "
100 PRINT
110 PRINT USING a$; x; y1(1); y1(2)
120 x1 = a: h = (b - a) / m
130 c(1) = 0: c(2) = .5: c(3) = .5: c(4) = 1
140 FOR i = 1 TO m
150 FOR j = 1 TO 4
160 FOR n = 1 TO 2
170 x = x1 + c(j) * h: y(n) = y1(n) + c(j) * k(j - 1, n)
180 NEXT n
190 FOR n = 1 TO 2
200 GOSUB 330
210 k(j, n) = h * f(n)
220 NEXT n
230 NEXT j
240 FOR n = 1 TO 2
250 y1(n) = y1(n) + (k(1,n) + 2*k(2,n) + 2*k(3,n) + k(4,n)) / 6
260 NEXT n
270 x1 = x1 + h
280 PRINT USING a$; x1; y1(1); y1(2)
290 NEXT i
300 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
310 IF INKEY$ = "" THEN 310
320 END
330 f(1) = y(2): f(2) = -y(1)
340 RETURN

```

3⁰. Программа численного решения задачи Коши для системы дифференциальных уравнений первого порядка методом Рунге - Кутты на языке PASCAL

```
program MethodOfRungeKutt;
```

```

( *****
РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ СИСТЕМЫ ДИФФЕРЕНЦИАЛЬНЫХ
УРАВНЕНИЙ ПЕРВОГО ПОРЯДКА МЕТОДОМ РУНГЕ - КУТТА
***** )

```

```

uses Crt;
const
  c:array[1..4] of real = (0,0.5,0.5,1);
type
  coef2 = array[0..4,1..2] of real;
  vect = array[1..2] of real;
var
  i,j,n,m:integer;
  a,b,h,x:real;
  y,y1:vect;
  k:coef2;
  ch:char;
{
  ПОДПРОГРАММЫ
}
{ Правая часть дифференциального уравнения - функция f(x,y) }
function f(i:integer;x:real;y:vect):real;
begin
  case 1 of
    1: f := y[2];
    2: f := -y[1];
  end;
end;
{
  процедура Pausa;
begin
  writeln; writeln ('Для продолжения нажмите любую клавишу...');
  repeat ch := readkey until ch <> '';
end;
}
{
  ОСНОВНАЯ ПРОГРАММА
}
begin
  clrscr;
  writeln ('Введите значения концов отрезка [a,b]');
  read (a, b);
  writeln ('! Введите начальные значения');
  writeln ('функции y10 и функции y20 при x=x0');
  read (y1[1], y1[2]);
  writeln ('Введите число значений функции на промежутке (a,b)');

```

```

READ (m);
  x := a; h := (b - a) / m;
  WRITELN ('x=',x:5:2,' y1=',y1[1]:9:6,' y2=',y1[2]:9:6);
  FOR i := 1 TO m DO
    BEGIN
      FOR j := 1 TO 4 DO
        BEGIN
          FOR n := 1 TO 2 DO
            y[n] := y1[n] + c[j] * k[j - 1, n];
          FOR n := 1 TO 2 DO
            k[j, n] := h * f(n, x + c[j] * h, y);
        END;
        FOR n := 1 TO 2 DO
          y1[n] := y1[n] + (k[1, n] + 2*k[2, n] + 2*k[3, n] + k[4, n])/6;
          x := x + h;
          WRITELN ('x=',x:5:2,' y1=',y1[1]:9:6,' y2=',y1[2]:9:6);
        END;
      PAUSA;
    END.
  END.

```

4⁰. Программа численного решения задачи Коши для системы дифференциальных уравнений первого порядка методом Рунге - Кутты на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
program MethodOfRungeKutt
C *****
C РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ СИСТЕМЫ ДИФФЕРЕНЦИАЛЬНЫХ
C УРАВНЕНИЙ ПЕРВОГО ПОРЯДКА МЕТОДОМ РУНГЕ - КУТТА
C *****
INTEGER*2 system
REAL k(0:4,2)
DIMENSION c(4),y(2),y1(2)
i=system('cls'C)
WRITE (*,*) 'Введите значения концов отрезка [a,b]'
READ (*,*) a,b
WRITE (*,*) 'Введите начальные значения '
WRITE (*,*) 'функции y10 и функции y20 при x=x0'

```

```

READ (*,*) y1(1),y1(2)
WRITE (*,*) 'Введите число шагов на отрезке [a,b]'
READ (*,*) m
  x = a
  h = (b - a) / m
WRITE (*, '(3(A5,E13.7))') ' x=',x,' y1=',y1(1),' y2=',y1(2)
c(1) = 0
c(2) = 0.5
c(3) = 0.5
c(4) = 1
DO i = 1,m
  DO j = 1,4
    DO n = 1,2
      y(n) = y1(n) + c(j) * k(j - 1, n)
    END DO
    DO n = 1,2
      k(j, n) = h * f(n, x + c(j) * h, y)
    END DO
  END DO
  DO n = 1,2
    y1(n) = y1(n) + (k(1,n) + 2*k(2, n) + 2*k(3,n) + k(4,n)) / 6
  END DO
  x = x + h
WRITE (*, '(3(A5,E13.7))') ' x=',x,' y1=',y1(1),' y2=',y1(2)
END DO
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

```

FUNCTION f (i, x, y)
DIMENSION y(2)
SELECT CASE (i)
  CASE (1)
    f = y(2)
  CASE (2)
    f = -y(1)
END SELECT
RETURN
END

```


**5⁰. Программа численного решения задачи Коши
для системы дифференциальных уравнений первого порядка
методом Рунге - Кутты на языке C**

/******

РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ СИСТЕМЫ ДИФФЕРЕНЦИАЛЬНЫХ
УРАВНЕНИЙ ПЕРВОГО ПОРЯДКА МЕТОДОМ РУНГЕ - КУТТА

*****/*

#include <stdio.h>

#include <conio.h>

/* -----

ПОДПРОГРАММА

Правые части дифференциальных уравнений */

float f(int i, float x, float y[3])

{ float z;

switch (i)

{ case 1: z = y[2]; break;

case 2: z = -y[1]; break;

};

return z;

}

/* -----

ОСНОВНАЯ ПРОГРАММА

*/

void main()

{ int i,j,n,m;

float a,b,h,x,y[3],y1[3],k[5][3];

float c[4]={0,.5,.5,1};

clrscr();

printf ("Введите значения концов отрезка [a,b]\n");

scanf ("%f%f",&a,&b);

printf ("Введите начальные значения n");

printf ("функции y10 и функции y20 при x=x0\n");

scanf ("%f%f",&y1[1],&y1[2]);

printf ("Введите число значений функции на промежутке (a,b)\n");

scanf ("%i",&m);

x = a; h = (b - a) / m;

```

printf ("x=%-5.2f y1=%9.6f y2=%9.6f\n",x,y1[1],y1[2]);
for (i = 1; i <= m; i++)
{ for (j = 1; j <= 4; j++)
  { for (n = 1; n <= 2; n++)
    y[n] = y1[n] + c[j-1] * k[j - 1][n];
    for (n = 1; n <= 2; n++)
      k[j][n] = h * f(n, x + c[j-1] * h, y);
  };
  for (n = 1; n <= 2; n++)
    y1[n] += (k[1][n]+2*k[2][n]+2*k[3][n]+k[4][n])/6;
  x += h;
printf ("x=%-5.2f y1=%9.6f y2=%9.6f\n",x,y1[1],y1[2]);
};
printf("\n\nНажмите любую клавишу для продолжения...");
getch();
}

```

6⁰. Программа численного решения задачи Коши для системы дифференциальных уравнений первого порядка методом Рунге - Кутты на языке QUICKBASIC

При решении, например, задачи Коши для системы двух дифференциальных уравнений второго порядка

$$\begin{cases} z_1'' + z_1' + 3(z_1 - z_2) = \sin x, \\ z_2'' + z_2' + 5(z_2 - z_1) = 0, \end{cases}$$

$$z_1|_{x=x_0}=z_{10}, \quad z_1'|_{x=x_0}=z'_{10}, \quad z_2|_{x=x_0}=z_{20}, \quad z_2'|_{x=x_0}=z'_{20}$$

ее можно свести к следующей задаче Коши для системы дифференциальных уравнений первого порядка:

$$\begin{cases} y_1' = y_2, \\ y_2' = -y_2 - 3(y_1 - y_3) + \sin x, \\ y_3' = y_4, \\ y_4' = -y_4 - 5(y_3 - y_1), \end{cases}$$

$$y_1|_{x=x_0}=z_{10}, \quad y_2|_{x=x_0}=z'_{10}, \quad y_3|_{x=x_0}=z_{20}, \quad y_4|_{x=x_0}=z'_{20}.$$

Численное решение последней задачи на сетке отрезка $[a, b]$ ре-

лизуется с помощью программы. Решение исходной задачи получается автоматически, если учесть, что $z_1(x) = y_1(x)$, а $z_2(x) = \dot{y}_3(x)$.

```
' *****
' РЕШЕНИЕ ЗАДАЧИ КОПИ ДЛЯ СИСТЕМЫ ДИФФЕРЕНЦИАЛЬНЫХ
' УРАВНЕНИЙ ПЕВОВОГО ПОРЯДКА МЕТОДОМ РУНГЕ - КУТТА
' *****

DECLARE SUB pausa ()
DECLARE FUNCTION f! (i%, x!, y!())
DEFINT I-J, M-N: DIM c(4), k(4, 4), y(4), y1(4)
CLS
PRINT " Введите через запятую"
PRINT "значения концов отрезка [a,b]"
INPUT a, b
PRINT "Введите через запятую начальные значения"
PRINT "функций y10, y20, y30, y40 при x=x0"
INPUT y1(1), y1(2), y1(3), y1(4)
PRINT "Введите число значений функции на промежутке (a,b)"
INPUT m
a1$ = "x = ##.## y1 = ##.##### y2 = ##.##### "
a2$ = "y3 = ##.##### y4 = ##.##### "
a$ = a1$ + a2$
PRINT USING a$; x; y1(1); y1(2); y1(3); y1(4)
    x = a: h = (b - a) / m
    c(1) = 0: c(2) = .5: c(3) = .5: c(4) = 1
    FOR i = 1 TO m
        FOR j = 1 TO 4
            FOR n = 1 TO 4
                y(n) = y1(n) + c(j) * k(j - 1, n)
            NEXT
            FOR n = 1 TO 4
                k(j, n) = h * f(n, x + c(j) * h, y())
            NEXT
        NEXT
    NEXT
    FOR n = 1 TO 4
        y1(n) = y1(n) + (k(1, n) + 2*k(2, n) + 2*k(3, n) + k(4, n)) / 6
    NEXT: x = x + h
```

```
PRINT USING a$: x; y1(1); y1(2); y1(3); y1(4)
NEXT
CALL pausa
END

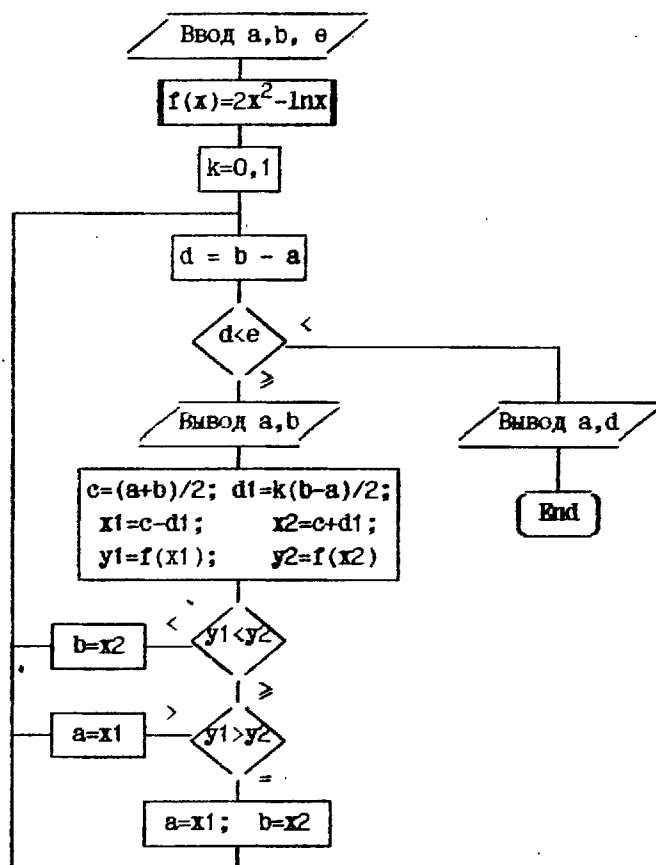
DEFSNG I-J, M-N
FUNCTION f (1%, x, y())
DEFINT I
SELECT CASE 1
CASE 1: f = y(2)
CASE 2: f = -y(2) - .3 * (y(1) - y(3)) + SIN(x)
CASE 3: f = y(4)
CASE 4: f = -y(4) - 5 * (y(3) - y(1))
END SELECT
END FUNCTION

DEFSNG I
SUB pausa
PRINT : PRINT "Для продолжения нажмите любую клавишу..."
DO
LOOP WHILE INKEY$ = ""
END SUB
```

К §3

1⁰. Блок-схема поиска минимума функции одной переменной
методом половинного деления

В блок-схеме и программах настоящего параграфа рассматривается унимодальная на отрезке $[a, b]$ функция $f(x) = 2x^2 - \ln x$ (см. пример 3 из §3). Для этой функции находится точка локального минимума с заданной точностью ϵ .



2⁰. Программа поиска минимума функции одной переменной методом половинного деления на языке BASIC

```

1 REM *****
2 REM МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ ДЛЯ ОПРЕДЕЛЕНИЯ
3 REM ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
4 REM *****
10 CLS
20 PRINT "Введите через запятую значения концов "
30 INPUT "отрезка [a,b] унимодальности функции f(x) "; a, b
40 INPUT "Задайте точность нахождения точки min f(x) "; e
50 DEF fnf (x) = 2 * x ^ 2 - LOG(x)
60 k = .1
70 d = b - a: GOSUB 210
80 IF d < e THEN 150
90 c = (a + b) / 2: d1 = k * (b - a) / 2
100 x1 = c - d1: x2 = c + d1
110 y1 = fnf(x1): y2 = fnf(x2)
120 IF y1 < y2 THEN b = x2: GOTO 70
130 IF y1 > y2 THEN a = x1: GOTO 70
140 a = x1: b = x2: GOTO 70
150 PRINT USING "Т. минимума x=##.##### погр.=#.#####"; a; d
160 GOSUB 180
170 END
171 REM -----
172 REM ПОДПРОГРАММЫ
173 REM -----
180 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
190 IF INKEY$ = "" THEN 190
200 RETURN
201 REM -----
210 CLS
220 PRINT "Отрезок унимодальности [a,b]"
230 PRINT USING "a =##.##### b =##.#####"; a; b
240 GOSUB 180
250 RETURN

```

3⁰. Программа поиска минимума функции одной переменной методом половинного деления на языке PASCAL

```

( *****
  МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ ДЛЯ ОПРЕДЕЛЕНИЯ
  ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
  ***** )

program FindMinOfFuncByDiv;
uses Crt;
const
  k = 0.1;
var
  a,b,c,d,d1,e,x1,x2,y1,y2:real;
  ch:char;
( -----
                                ПОДПРОГРАММЫ                                )

procedure PAUSA;
begin
  WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
  REPEAT ch := readkey UNTIL ch <> '';
  END;
( ----- )

function f(x:real):real;
begin
  f := 2*x*x - ln(x);
  END;
( ----- )

                                ОСНОВНАЯ ПРОГРАММА                                )

begin
  clrscr;
  WRITELN ('      Введите значения концов');
  WRITELN ('отрезка [a,b] унимодальности функции f(x)');
  READ (a, b);
  WRITELN ('Задайте точность нахождения точки min f(x)');
  READ (e);
  REPEAT
    d := b - a;

```

```

ClrScr;
WRITELN ('Отрезок унимодальности [a,b]');
WRITELN ('a = ',a:9:6,' b = ',b:9:6);
Pausa;
c := (a + b) / 2; d1 := k * (b - a) / 2;
x1 := c - d1; x2 := c + d1;
y1 := f(x1); y2 := f(x2);
IF y1 < y2 THEN b := x2;
IF y1 > y2 THEN a := x1;
IF y1 = y2 THEN
  BEGIN
    a := x1; b := x2;
  END;
UNTIL d < e;
WRITELN ('Т. минимума x = ',a:9:6,' погр. = ',d:9:6);
Pausa;
END.

```

4⁰. Программа поиска минимума функции одной переменной методом половинного деления на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
C *****
C МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ ДЛЯ ОПРЕДЕЛЕНИЯ
C ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
C *****
program FindMinOfFuncByDiv
INTEGER*2 system
REAL k
f (x) = 2 * x * x - LOG(x)
i=system('cls'C)
WRITE (*,*) ' Введите значения концов'
WRITE (*,*) 'отрезка [a,b] унимодальности функции f(x)'
READ (*,*) a, b
WRITE (*,*) 'Задайте точность нахождения точки min f(x)'
READ (*,*) e

```



```

WRITE (*,*)
k = .1
d = 1
DO WHILE (d.GE.e)
    d = b - a
    i=system('cls'C)
    WRITE (*,*) 'Отрезок унимодальности [a,b]'
    WRITE (*, '(A4,F9.6,A5,F9.6)') ' a =',a,' b =', b
    PAUSE 'Нажмите клавишу ENTER для продолжения...'
    c = (a + b) / 2
    d1 = k * (b - a) / 2
    x1 = c - d1
    x2 = c + d1
    y1 = f(x1)
    y2 = f(x2)
    IF (y1.LT.y2) b = x2
    IF (y1.GT.y2) a = x1
    IF (y1.EQ.y2) THEN
        a = x1
        b = x2
    END IF
END DO
WRITE (*, '(A10,F9.6,A7,F9.6)') ' Т. мин x =',a,' погр. =',d
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

**5⁰. Программа поиска минимума функции одной переменной
методом половинного деления на языке C**

```

/*
*****
МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ ДЛЯ ОПРЕДЕЛЕНИЯ
ТОЧКИ МИНИМУМА ДЛЯ ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
*****
#include <stdio.h>
#include <conio.h>
#include <math.h>

```

```
#define K 0.1
```

```
/* -----  
                                ПОДПРОГРАММЫ                                */
```

```
float f(float x)  
{ float y; y = 2*x*x - log(x); return y; }
```

```
/* -----*/
```

```
void pausa()
```

```
{ printf("\n\nНажмите любую клавишу для продолжения...");  
  getch();  
}
```

```
/* -----  
                                ОСНОВНАЯ ПРОГРАММА                                */
```

```
void main()
```

```
{ float a,b,c,d,d1,e,x1,x2,y1,y2;  
  clrscr();  
  printf ("          Введите значения концов n");  
  printf ("отрезка [a,b] унимодальности функции f(x)\n");  
  scanf ("%f%f",&a,&b);  
  printf ("Задайте точность нахождения точки min f(x)\n");  
  scanf ("%f",&e);
```

```
do
```

```
{ d = b - a;  
  clrscr();  
  printf ("Отрезок унимодальности [a,b]\n");  
  printf ("a = %f b = %f",a,b);  
  pausa();
```

```
  c = (a + b) / 2; d1 = K * (b - a) / 2;
```

```
  x1 = c - d1; x2 = c + d1;
```

```
  y1 = f(x1); y2 = f(x2);
```

```
  if (y1 < y2) b = x2;
```

```
  if (y1 > y2) a = x1;
```

```
  if (y1 == y2) { a = x1; b = x2; };
```

```
}
```

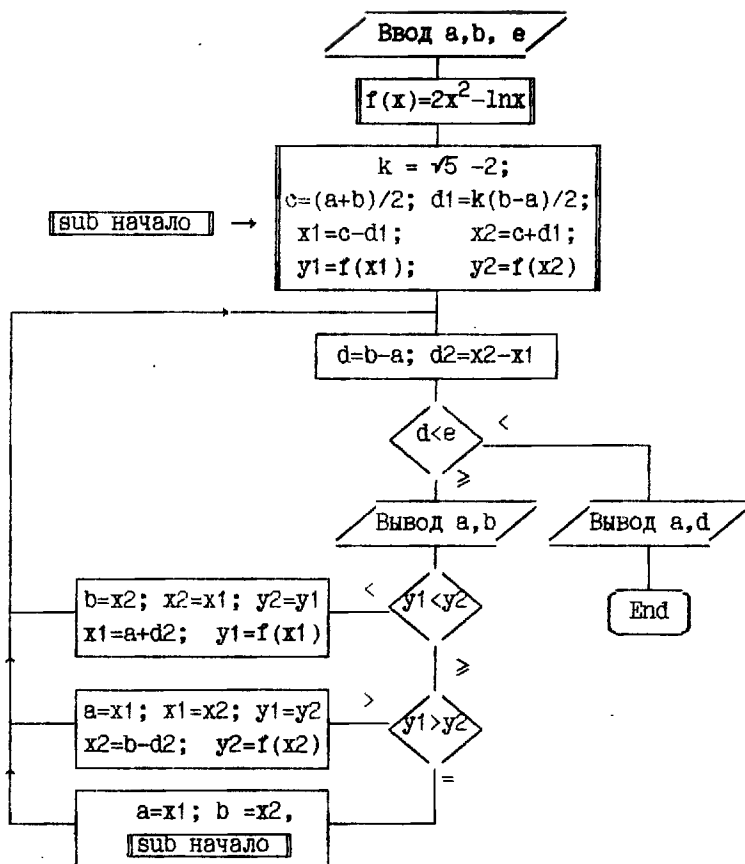
```
while (d >= e);
```

```
printf ("\n\nТочка минимума x = %f погрешн. = %f",a,d);
```

```
pausa();
```

```
}
```

1¹. Блок-схема поиска минимума функции одной переменной методом золотого сечения



2¹. Программа поиска минимума функции одной переменной методом золотого сечения на языке BASIC

```

1 REM *****
2 REM      МЕТОД      ЗОЛОТОГО СЕЧЕНИЯ      ПОИСКА
3 REM      ТОЧКИ МИНИМУМА      ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
4 REM      *****

```

```

10 DEFDBL A - F, K, X - Y: CLS
20 PRINT "Введите через запятую значения концов "
30 INPUT "отрезка [a,b] унимодальности функции f(x) "; a, b
40 INPUT "Задайте точность нахождения точки min f(x) "; e
50 DEF fnf (x) = 2 * x ^ 2 - LOG(x)
60 k = SQR(5) - 2: GOSUB 230
70 d = b - a: d2 = x2 - x1: GOSUB 180
80 IF d < e THEN 120
90 IF y1 < y2 THEN b=x2: x2=x1: y2=y1: x1=a+d2: y1=fnf(x1): GOTO 70
100 IF y1 > y2 THEN a=x1: x1=x2: y1=y2: x2=b-d2: y2=fnf(x2): GOTO 70
110 a = x1: b = x2: GOSUB 230: GOTO 70
120 PRINT USING "T. минимума X=###.##### погр.=#.#####"; a; d
130 GOSUB 150
140 END
141 REM -----
142 REM ПОДПРОГРАММЫ
143 REM -----
150 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
160 IF INKEY$ = "" THEN 160
170 RETURN
171 REM -----
180 CLS
190 PRINT "Отрезок унимодальности [a,b]"
200 PRINT USING "a =###.##### b =###.#####"; a; b
210 GOSUB 150
220 RETURN
230 c = (a + b) / 2: d1 = k * (b - a) / 2
240 x1 = c - d1: x2 = c + d1
250 y1 = fnf(x1): y2 = fnf(x2)
260 RETURN

```

3¹. Программа поиска минимума функции одной переменной методом золотого сечения на языке PASCAL

```

{
*****
    МЕТОД ЗОЛОТОГО СЕЧЕНИЯ ПОИСКА
    ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
*****}
program FindMinOfFuncByGoldSec;

```

```

uses Crt;
var
a,b,d,d2,e,x1,x2,y1,y2:real;
    ch:char;
{ -----
                                ПОДПРОГРАММЫ                                }

procedure PAUSA;
begin
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> '';
    END;
{ ----- }

function f(x:real):real;
begin
    f := 2*x*x - Ln(x);
    END;
{ ----- }

procedure frombeginning;
var
    k,c,d1:real;
begin
    k := SQRT(5) - 2;
    c := (a + b) / 2; d1 := k * (b - a) / 2;
    x1 := c - d1; x2 := c + d1;
    y1 := f(x1); y2 := f(x2);
    END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                }

BEGIN
    ClrScr;
    WRITELN ('          Введите значения концов');
    WRITELN ('отрезка [a,b] унимодальности функции f(x)');
    READ (a, b);
    WRITELN ('Задайте точность нахождения точки min f(x)');
    READ (e);
    frombeginning;
    REPEAT
        d := b - a; d2 := x2 - x1;

```

```

ClrScr;
WRITELN ('Отрезок унимодальности [a,b]');
WRITELN ('a = ',a:9:6,' b = ',b:9:6);
Pausa;
IF y1 < y2 THEN
  BEGIN
    b := x2; x2 := x1; y2 := y1; x1 := a + d2; y1 := f(x1);
  END
ELSE
  BEGIN
    IF y1 > y2 THEN
      BEGIN
        a := x1; x1 := x2; y1 := y2; x2 := b - d2; y2 := f(x2);
      END
    ELSE
      BEGIN
        a := x1; b := x2; frombeginning;
      END;
    END;
  END;
UNTIL d < e;
WRITELN ('Т. минимума x = ',a:9:6,' погр. = ',d:9:6);
Pausa;
END.

```

4¹. Программа поиска минимума функции одной переменной методом золотого сечения на языке FORTRAN

```

$INCLUDE: 'EXEC.PI'
C *****
C          МЕТОД ЗОЛОТОГО СЕЧЕНИЯ    ПОИСКА
C          ТОЧКИ МИНИМУМА  ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
C          *****
program FindMinOfFuncByGoldSec
INTEGER*2 system
COMMON a,b,x1,x2,y1,y2
i=system('cls'C)
WRITE (*,*) '      Введите значения концов'

```

```

WRITE (*,*) 'отрезка [a,b] унимодальности функции f(x)'
READ (*,*) a, b
WRITE (*,*) 'Задайте точность нахождения точки min f(x)'
READ (*,*) e
WRITE (*,*)
CALL frombeginning
d = 1
DO WHILE (d.GE.e)
    d = b - a
    d2 = x2 - x1
    i=system('cls'0)
    WRITE (*,*) 'Отрезок унимодальности [a,b]'
    WRITE (*, '(A4,F9.6,A5,F9.6)') ' a =',a,' b =', b
    PAUSE 'Нажмите клавишу ENTER для продолжения...'
    IF (y1.LT.y2) THEN
        b = x2
        x2 = x1
        y2 = y1
        x1 = a + d2
        y1 = f(x1)
    ELSE
        IF (y1.GT.y2) THEN
            a = x1
            x1 = x2
            y1 = y2
            x2 = b - d2
            y2 = f(x2)
        ELSE
            a = x1
            b = x2
            CALL frombeginning
        END IF
    END IF
END DO
WRITE (*, '(A10,F9.6,A7,F9.6)') ' Т. мин x=',a,' погр.=' ,d
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END
FUNCTION f(x)

```

```

      f = 2*x*x - LOG(x)
RETURN
END
SUBROUTINE frombeginning
REAL k
COMMON a,b,x1,x2,y1,y2
k = SQRT(5) -2
  c = (a + b) / 2
  d1 = k * (b - a) / 2
  x1 = c - d1
  x2 = c + d1
  y1 = f(x1)
  y2 = f(x2)
RETURN
END

```

5¹. Программа поиска минимума функции одной переменной методом золотого сечения на языке C

```

/* *****
      МЕТОД ЗОЛОТОГО СЕЧЕНИЯ ДЛЯ ПОИСКА
      ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
      *****/
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define K (sqrt(5)-2)
float a,b,x1,x2,y1,y2;
/* -----
                        ПОДПРОГРАММЫ                                */
float f(float x)
{ float y; y = 2*x*x - log(x); return y; }
/* -----*/
void pausa()
{ printf("\n\nНажмите любую клавишу для продолжения...");
  getch();
}

```



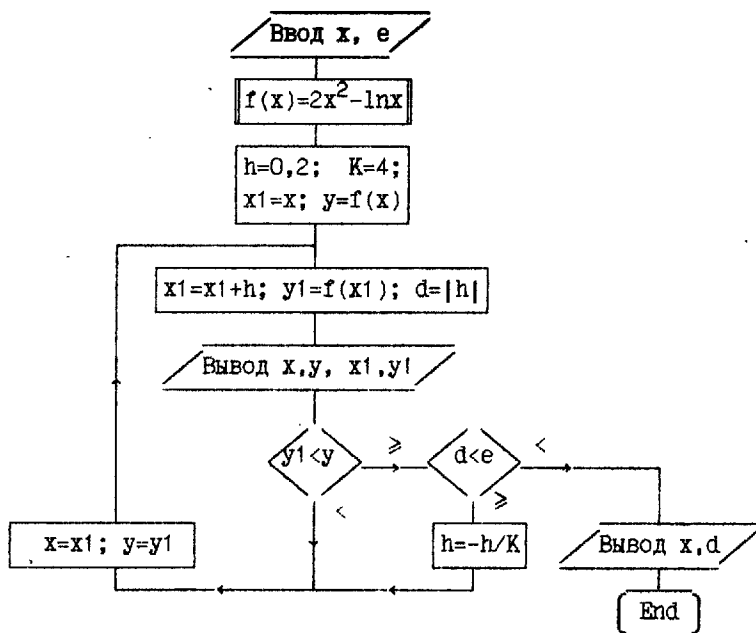
```

/* ----- */
void frombeginning()
{ float c,d1;
  c = (a + b) / 2; d1 = K * (b - a) / 2;
  x1 = c - d1; x2 = c + d1;
  y1 = f(x1); y2 = f(x2);
}
/* ----- */
                                ОСНОВНАЯ ПРОГРАММА                                */

void main()
{ float d,d2,e;
  clrscr();
  printf ("          Введите значения концов n");
  printf ("отрезка [a,b] унимодальности функции f(x)\n");
  scanf ("%f%f",&a,&b);
  printf ("Задайте точность нахождения точки min f(x)\n");
  scanf ("%f",&e);
  frombeginning();
do
{ d = b - a; d2 = x2 - x1;
  clrscr();
  printf ("Отрезок унимодальности [a,b]\n");
  printf ("a = %f b = %f",a,b);
  pausa();
  if (y1 < y2)
  { b = x2; x2 = x1; y2 = y1; x1 = a + d2; y1 = f(x1);}
  else
  { if (y1 > y2)
    { a = x1; x1 = x2; y1 = y2; x2 = b - d2; y2 = f(x2);}
    else
    { a = x1; b = x2; frombeginning();}
  };
}
while (d >= e);
printf ("\n\nТочка минимума x = %f погрешн. = %f",a,d);
pausa();
}

```

1². Блок-схема поиска минимума функции одной переменной методом сканирования



2². Программа поиска минимума функции одной переменной методом сканирования на языке BASIC

```

1 REM *****
2 REM ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
3 REM МЕТОДОМ СКАНИРОВАНИЯ
4 REM *****
10 CLS
20 INPUT "Введите некоторое значение x > 0 "; x
30 INPUT "Задайте точность нахождения точки min f(x) "; e
40 DEF fnf (x) = 2 * x ^ 2 - LOG(x)
50 h = .2: K = 4
60 x1 = x: y = fnf(x): GOTO 80
70 x = x1: y = y1

```



```

BEGIN
WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
REPEAT ch := readkey UNTIL ch <> '';
END;
{ ----- }
function f(x:real):real;
BEGIN
  f := 2*x*x - ln(x);
END;
{ ----- }
                                ОСНОВНАЯ ПРОГРАММА                                }
BEGIN
  ClrScr;
  WRITELN ('Введите некоторое значение  $x > 0$  ');
  READ (x);
  WRITELN ('Задайте точность нахождения точки  $\min f(x)$  ');
  READ (e);
  h := 0.2; x1 := x; y := f(x);
  REPEAT
    d := ABS(h); x1 := x1 + h; y1 := f(x1); a := (y1 >= y);
  { ----- Вывод на экран промежуточных вычислений ----- }
    ClrScr;
    WRITELN ('Два последовательных значения аргумента x, x1');
    WRITELN ('x = ',x:9:6,' x1 = ',x1:9:6);
    WRITELN ('y = ',y:9:6,' y1 = ',y1:9:6);
    Pausa;
  { ----- }
    IF a THEN h := - h / K;
    x := x1; y := y1;
  UNTIL a AND (d<e);
  x := x + h * K;
  WRITELN ('Т. минимума x = ',x:9:6,' погр. = ',d:9:6);
  Pausa;
END.

```

4². Программа поиска минимума функции одной переменной методом сканирования на языке FORTRAN

```
$INCLUDE: 'EKES.PI'
```

```

C      *****
C      ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
C      МЕТОДОМ СКАНИРОВАНИЯ
C      *****

program FindMinOfFuncByScanMeth
  INTEGER*2 system
  LOGICAL a
  f(x) = 2*x*x - LOG(x)
  i=system('cls'C)
  WRITE (*,*) 'Введите некоторое значение x > 0'
  READ (*,*) x
  WRITE (*,*) 'Задайте точность нахождения точки min f(x)'
  READ (*,*) e
  WRITE (*,*)
  h = .2
  K = 4
  y = f(x)
  x1 = x
  d = 1
  DO WHILE (a.OR.(d.GE.e))
    x1 = x1 + h
    y1 = f(x1)
    a = (y1.LT.y)
    d = ABS(h)
C  ----- Вывод на экран промежуточных вычислений -----
    i=system('cls'C)
    WRITE (*,*) 'Отрезок унимодальности [a,b]'
    WRITE (*, '(A4,F9.6,A6,F9.6)') ' x =', x, ' x1 =', x1
    WRITE (*, '(A4,F9.6,A6,F9.6)') ' y =', y, ' y1 =', y1
    PAUSE 'Нажмите клавишу ENTER для продолжения...'
C  -----
    IF (.NOT.a) h = -h/K
    x = x1
    y = y1
  END DO

```

```

X = X + h*K
WRITE (*, '(A10,P9.6,A7,P9.6)') ' Т. мин X=', X, ' погр.= ', d
PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

5². Программа поиска минимума функции одной переменной методом сканирования на языке C

```

/* *****
ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ
МЕТОДОМ СКАНИРОВАНИЯ
***** */

#include <stdio.h>
#include <conio.h>
#include <math.h>
#define K 4

/* -----
ПОДПРОГРАММЫ
*/

float f(float x)
{ float y; y = 2*x*x - log(x); return y; }

/* ----- */

void pausa()
{ printf("\n\nНажмите любую клавишу для продолжения...\n");
  getch();
}

/* -----
ОСНОВНАЯ ПРОГРАММА
*/

void main()
{ short a;
  float d,e,h,x,x1,y,y1;
  clrscr();
  printf ("Введите некоторое значение x > 0\n");
  scanf ("%f",&x);
  printf ("Задайте точность нахождения точки min f(x)\n");
  scanf ("%f",&e);
  h = 0.2; x1 = x; y = f(x);
  do
  { d = fabs(h); x1 += h; y1 = f(x1); a = (y1 < y);

```

```

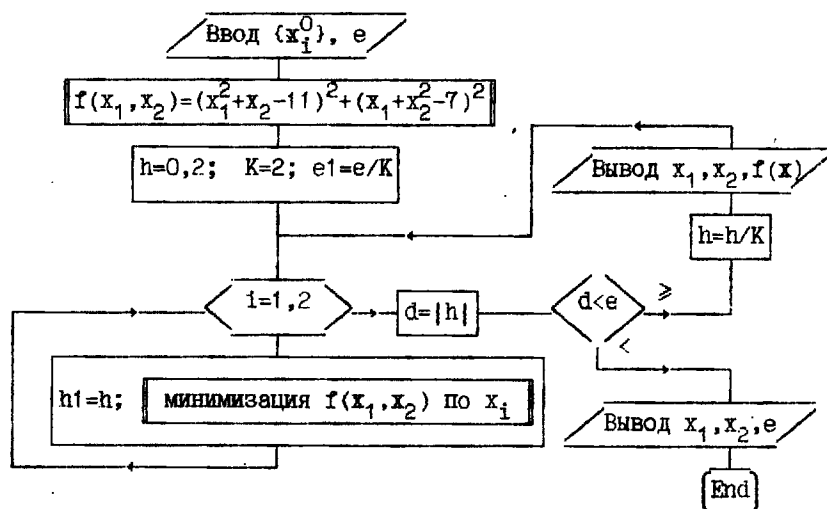
/* --- Вывод на экран промежуточных вычислений --- */
clrscr();
printf ("Два последовательных значения");
printf (" аргумента x, x1\n");
printf ("x = %f x1 = %f\n",x,x1);
printf ("y = %f y1 = %f\n",y,y1);
pausa();

/* ----- */
if (!a) h /= - K;
x = x1; y = y1;
}
while (a || d >= e);
x += h * K;
printf ("\nТочка минимума x = %f погрешность = %f",x,d);
pausa();
}

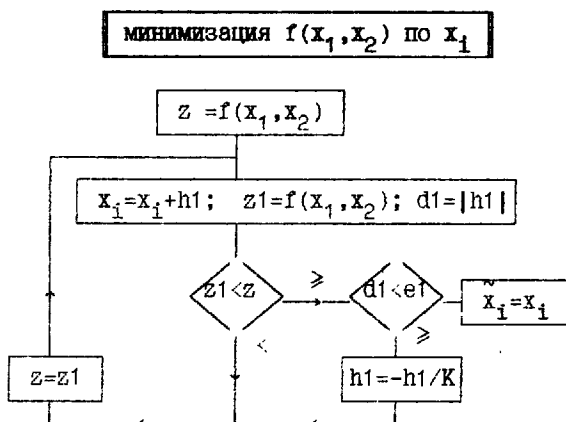
```

К §4

- 1⁰. Блок-схема поиска минимума функции двух переменных методом покоординатного спуска



ПОДПРОГРАММА ОДНОМЕРНОЙ МИНИМИЗАЦИИ
МЕТОДОМ СКАНИРОВАНИЯ ПО ПЕРЕМЕННОЙ x_1



В блок-схеме и программах рассматривается функция Химмельблау $f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$, для которой находятся точки локального минимума.

Вычисления по программам приведены в таблицах к примеру 1 (см. с. 176).

Структура программ допускает их естественное обобщение для поиска локального минимума функции трех и более переменных. Условие (14) прекращения вычислительной процедуры при достижении заданной точности ϵ заменено практически равноценным, особенно при достижении высокой точности, условием $|h| < \epsilon$. Шаг h — стартовое изменение переменной в задаче одномерной минимизации, уменьшающееся от одного этапа вычислений к другому в K раз (целое число $K \geq 2$). Одномерная минимизация осуществляется с уменьшающимся шагом h_1 до выполнения условия $|h_1| < \epsilon_1$. Точность ϵ_1 в одномерной задаче принята несколько более высокой по сравнению с ϵ : $\epsilon_1 = \epsilon/K$. Если положить h отличным от принятого в программе $h = 0.2$ и изменить начальный вектор $\mathbf{x}^{(0)}$, то вычисления по программе снова приведут к одной из четырех точек локального минимума функции Химмельблау.

2⁰. Программа поиска минимума функции двух переменных методом покоординатного спуска на языке BASIC

```

1 REM *****
2 REM ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
3 REM МЕТОДОМ ПОКООРИНАТНОГО СПУСКА
4 REM *****
10 DEF fnf (x1, x2) = (x1^2 + x2 - 11)^2 + (x1 + x2^2 - 7)^2
20 DEFINT I: DIM x(2)
30 CLS
40 PRINT "Введите координаты начального вектора (x1,x2)"
50 FOR i = 1 TO 2
60 INPUT x(i)
70 NEXT i
80 INPUT "Задайте точность нахождения точки min f(x1,x2) "; e
90 h = .2: K = 2: : e1 = e / K: d = ABS(h)
100 GOSUB 260
110 FOR i = 1 TO 2
120 h1 = h: GOSUB 200
130 NEXT i
140 d = ABS(h): IF d > e THEN GOSUB 260: h = h / K: GOTO 110
150 a$ = "Т. минимума x1=##.##### x2=##.##### погр.=#.#####"
160 PRINT : PRINT USING a$; x(1); x(2); e
170 PRINT USING "f(x1,x2) =###.#####"; fnf(x(1), x(2))
180 GOSUB 330
190 END
191 REM -----
192 REM ПОДПРОГРАММЫ
193 REM -----
200 z = fnf(x(1), x(2)): GOTO 220
210 z = z1
220 x(1) = x(1) + h1: z1 = fnf(x(1), x(2)): d1 = ABS(h1)
230 IF z1 < z THEN GOTO 210
240 IF d1 > e1 THEN h1 = -h1 / K: GOTO 210
250 RETURN
251 REM -----
260 CLS
270 PRINT "Вектор приближения (x1,x2) на данном шаге вычисления"

```

```

280 b$ = "x1=##.##### x2 =##.##### "
290 PRINT USING b$; x(1); x(2)
300 PRINT USING "f(x1,x2) =#####.#####"; fnf(x(1), x(2))
310 GOSUB 330
320 RETURN
321 REM -----
330 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
340 IF INKEY$ = "" THEN 340
350 RETURN

```

3⁰. Программа поиска минимума функции двух переменных методом покоординатного спуска на языке PASCAL

```

program MinOfFuncByCoordDescent;
{ *****
  ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
    МЕТОДОМ ПОКООРДИНАТНОГО СПУСКА
  ***** }
uses Crt;
const
  n = 2; K=2;
type
  vector=array[1..n] of real;
var
  i:integer;
  d,e,e1,h,h1,z:real;
  x:vector;
  ch:char;
{ -----
                                ПОДПРОГРАММЫ                                }
procedure PAUSA;
  BEGIN
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');
    REPEAT ch := readkey UNTIL ch <> ' ';
  END;
{ ----- }
function f(x:vector):real;
  var a,b:real;

```

```

BEGIN
  a := x[1]*x[1] + x[2] - 11; b := x[1] + x[2]*x[2] - 7;
  f := a*a + b*b;
END;
{ ----- }
procedure ScanForOneDim (i:integer);
var
  a:boolean;
  d1,z1:real;
BEGIN
  z := f(x);
  REPEAT
    d1 := ABS(h1); x[1] := x[1] + h1; z1 := f(x); a := (z1 >= z);
    IF a THEN h1 := - h1 / K;
    z := z1;
  UNTIL a AND (d1 <= e1);
END;
{ ----- Вывод на экран промежуточных вычислений ----- }
procedure OutputResult;
BEGIN
  ClrScr;
  WRITELN ('Вектор приближения (x1,x2) на данном шаге вычислений');
  WRITELN ('x1 = ',x[1]:9:6,' x2 = ',x[2]:9:6);
  WRITELN ('f(x1,x2) = ',f(x):9:6);
  Pausa;
END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                ----- }
BEGIN
  ClrScr;
  WRITELN ('Введите координаты начального вектора (x1,x2)');
  FOR i := 1 TO n DO READ (x[i]);
  WRITELN ('Задайте точность нахождения точки min f(x)');
  READ (e);
  h := 0.2; e1 := e / K; OutputResult;
  REPEAT
    d := ABS(h);
    FOR i := 1 TO n DO

```

```

BEGIN
  h1 := h; ScanForOneDim (1);
END;
OutputResult;
h := h / K;
UNTIL d < e;
WRITELN ('Т. минимума x1 = ', x[1]:9:6, ' x2 = ', x[2]:9:6);
WRITELN ('Погрешность = ', e:9:6);
Pausa;
END.

```

4⁰. Программа поиска минимума функции двух переменных
методом покоординатного спуска на языке FORTRAN

```

$INCLUDE: 'EXEC.FI'
C *****
C ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
C МЕТОДОМ ПОКООРДИНАТНОГО СПУСКА
C *****
program MinOfFuncByCoordDeacent
  INTEGER*2 system
  COMMON x(2), K, e1, h1
  i=system('cls'C)
  WRITE (*,*) 'Введите координаты вектора (x1,x2)'
  READ (*,*) (x(1), i=1,2)
  WRITE (*,*) 'Задайте точность нахождения точки min f(x)'
  READ (*,*) e
  WRITE (*,*)
  h = 0.2
  K = 2
  e1 = e / K
  d = 1
  i=system('cla'C)
  CALL OutputResult
  DO WHILE (d.GE.e)
    d = ABS(h)
    DO i=1,2
      h1 = h

```

```

      CALL ScanForOneDIM (1)
    END DO
    i=system('cls'C)
    CALL OutputResult
    h = h/K
  END DO
  WRITE (*, '(A8,E12.6,A4,E12.6)') ' Min x1=', x(1), ' x2=', x(2)
  WRITE (*, '(A14,E12.6)') ' Погрешность =', e
  PAUSE 'Нажмите клавишу ENTER для продолжения...'
END

```

C

C

C

ПОДПРОГРАММЫ

```

FUNCTION f(x)
  DIMENSION x(2)
  f = (x(1)**2 + x(2) - 11)**2 + (x(1) + x(2)**2 - 7)**2
  RETURN
END

```

C

```

SUBROUTINE ScanForOneDIM (1)
  COMMON x(2), K, e1, h1
  LOGICAL a
  z = f(x)
  d1=1
  DO WHILE (a.OR.(d1.GE.e1))
    x(i) = x(i) + h1
    z1 = f(x)
    d1 = ABS(h1)
    a = (z1.LT.z)
    IF (.NOT.a) h1 = -h1/K
    z = z1
  END DO
  RETURN
END

```

C

```

----- Вывод на экран промежуточных вычислений -----
SUBROUTINE OutputResult
  COMMON x(2)
  WRITE (*,*) 'Вектор приближения (x1,x2) на шаге вычислений'

```

```

WRITE (*,'(A5,E12.6,A6,E12.6)') ' x1 =',x(1),' x2 =',x(2)
WRITE (*,'(A11,E12.6)') ' f(x1,x2) =',f(x)
PAUSE 'Нажмите клавишу ENTER для продолжения...'
RETURN
END

```

5⁰. Программа поиска минимума функции двух переменных методом покоординатного спуска на языке C

```

/* *****
ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
МЕТОДОМ ПОКООРИНАТНОГО СПУСКА
***** */
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define N 2
#define K 2
int i;
float e1,h,h1,z,x[N+1];
/* -----
                                ПОДПРОГРАММЫ                                */
float f(float x[N+1])
{ float a,b,y; a = x[1]*x[1]+x[2]-11; b=x[1]+x[2]*x[2]-7;
  y = a*a + b*b; return y; }
/* ----- */
void scanforonedim (i)
int i;
{ short a;
  float d1,z1;
  z = f(x);
  do
  { d1 = fabs(h1); x[i] += h1; z1 = f(x); a = (z1 < z);
    if (!a) h1 /= - K;
    z = z1;
  }
  while (a || d1 >= e1);
}

```

```

/* ----- */
void pausa()
{printf ("\nНажмите любую клавишу для продолжения...\n");
 getch();
}
/* ----- Вывод на экран промежуточных вычислений ----- */
void outputresult(void)
{ clrscr();
 printf ("Вектор приближения (x1,x2) на данном шаге");
 printf (" вычислений n");
 printf ("x1 = %f x2 = %f\n",x[1],x[2]);
 printf ("f(x1,x2) = %f\n",f(x));
 pausa();
}
/* -----
                                ОСНОВНАЯ ПРОГРАММА                                */
void main()
{ float d,e;
  clrscr();
  printf ("Введите координаты");
  printf (" начального вектора (x1,x2)\n");
  for (i = 1; i <= N; i++) scanf ("%f",&x[i]);
  printf ("Задайте точность нахождения точки min f(x)\n");
  scanf ("%f",&e);
  outputresult(); h = 0.2; e1 = e / K;
  do
  { d = fabs(h);
    for (i = 1; i <= N; i++)
      { h1 = h; scanforonedim (i); };
    outputresult();
    h /= K;
  }
  while (d >= e);
  printf ("Точка минимума x1 = %f x2 = %f\n",x[1],x[2]);
  printf ("Погрешность = %f\n",d);
  pausa();
}

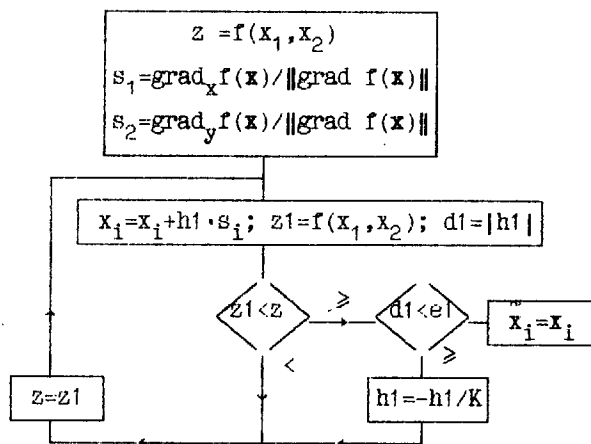
```

1¹. Блок-схема поиска минимума функции двух переменных методом скорейшего спуска

Для поиска минимума функции двух переменных методом скорейшего спуска можно использовать структуру программы поиска минимума функции двух переменных методом покоординатного спуска. Следует изменить содержание подпрограммы одномерной минимизации. Метод сканирования в задаче одномерной минимизации используется для поиска минимума функции двух переменных вдоль направления градиента функции Химмельблау (см. пример 2 §4). Структуру программы можно использовать для задач поиска точек минимума функции трех и более переменных.

*Подпрограмма одномерной минимизации
методом сканирования вдоль направления градиента функции*

минимизация $f(x_1, x_2)$ вдоль $S = \text{grad } f(x) / \|\text{grad } f(x)\|$



2¹. Программа поиска минимума функции двух переменных методом скорейшего спуска на языке BASIC

```

1 REM *****
2 REM ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
3 REM МЕТОДОМ СКОРЕЙШЕГО СПУСКА
4 REM *****

```



```

10 DEFINT I: DIM x(2), s(2)
20 DEF fnf(x1,x2)=(x1 ^ 2 + x2 - 11)^2 + (x1 + x2 ^ 2 - 7)^2
30 DEF fng1(x1,x2)=4*x1^3 + 4*x1*x2 + 2*x2^2 - 42*x1 - 14
40 DEF fng2(x1,x2)=4*x2^3 + 4*x1*x2 + 2*x1^2 - 26*x2 - 22
50 DEF fnd2 (x1, x2) = SQR(fng1(x1, x2) ^ 2 + fng2(x1, x2) ^ 2)
60 DEF fns1 (x1, x2) = fng1(x1, x2) / fnd2(x1, x2)
70 DEF fns2 (x1, x2) = fng2(x1, x2) / fnd2(x1, x2)
80 CLS
90 PRINT "Введите координаты начального вектора (x1,x2)"
100 FOR i = 1 TO 2
110     INPUT x(i)
120 NEXT i
130 IF fnd2(x(1), x(2)) <> 0 THEN 160
140     a$ = "Стационарная точка x1 = ##.#### x2 = ##.####"
150     PRINT USING a$; x(1); x(2): GOSUB 410: GOTO 250
160 INPUT "Задайте точность нахождения точки min f(x1,x2) "; e
170 h = .2: K = 2: e1 = e / K
180 GOSUB 360
190 GOSUB 260
200 d = ABS(h): IF d > e THEN h = h / K: GOSUB 360: GOTO 190
210 a$ = "Т. минимума x1=##.##### x2=##.##### погр.=##.#####"
220 PRINT : PRINT USING a$; x(1); x(2); e
230 PRINT USING "f(x1,x2) =###.#####"; fnf(x(1), x(2))
240 GOSUB 430
250 END
251 REM -----
252 REM ПОДПРОГРАММЫ
253 REM -----
260 s(1) = fns1(x(1), x(2)): s(2) = fns2(x(1), x(2))
270 z = fnf(x(1), x(2)): h1 = h: GOTO 290
280 z = z1
290 FOR i = 1 TO 2
300     x(i) = x(i) + h1 * s(i)
310 NEXT i
320 z1 = fnf(x(1), x(2)): d1 = ABS(h1)
330 IF z1 < z THEN GOTO 280
340 IF d1 > e1 THEN h1 = -h1 / K: GOTO 280
350 RETURN

```

```

351 REM -----
360 CLS
370 PRINT "Вектор приближения (x1,x2) на данном шаге вычисления"
380 b$ = "x1=##.##### x2 =##.##### "
390 PRINT USING b$; x(1); x(2)
400 PRINT USING "f(x1,x2) =#####.#####"; fnf(x(1), x(2))
410 GOSUB 430
420 RETURN
421 REM -----
430 PRINT : PRINT "Для продолжения нажмите любую клавишу..."
440 IF INKEY$ = "" THEN 440
450 RETURN

```

3¹. Программа поиска минимума функции двух переменных методом скорейшего спуска на языке PASCAL

```

program MinOfFuncByMethodOfGrad;
{
    *****
    ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
    МЕТОДОМ СКОРЕЙШЕГО СПУСКА
    *****
}
uses Crt;
label
    finish;
const
    n = 2; K=2;
type
    vector=array[1..n] of real;
var
    i:integer;
    absgrad,d,e,e1,h,h1,z:real;
    s,x:vector;
    ch:char;
{ -----
                                ПОДПРОГРАММЫ
                                }
procedure PAUSA;
    BEGIN
    WRITELN; WRITELN ('Для продолжения нажмите любую клавишу...');

```

```

REPEAT ch := readkey UNTIL ch <> '';
END;
{ ----- }
function f(x:vector):real;
  var a,b:real;
  BEGIN
    a := x[1]*x[1] + x[2] - 11; b := x[1] + x[2]*x[2] - 7;
    f := a*a + b*b;
  END;
{ ----- }
procedure GradDirection;
  var
    gradx,grady:real;
  BEGIN
    gradx := 4*x[1]*x[1]*x[1]+4*x[1]*x[2]+2*x[2]*x[2]-42*x[1]-14;
    grady := 4*x[2]*x[2]*x[2]+4*x[1]*x[2]+2*x[1]*x[1]-26*x[2]-22;
    absgrad := SQRT(gradx*gradx + grady*grady);
    IF absgrad <> 0 THEN
      BEGIN
        s[1] := gradx / absgrad;
        s[2] := grady / absgrad;
      END;
    END;
  { ----- }
procedure ScanForOneDim;
  var
    a:boolean;
    d1,z1:real;
  BEGIN
    z := f(x); GradDirection;
  REPEAT
    d1 := ABS(h1);
    FOR i := 1 TO n DO x[i] := x[i] + h1*s[i];
    z1 := f(x); a := (z1 >= z);
    IF a THEN h1 := - h1 / K;
    z := z1;
  UNTIL a AND (d1<e1);
END;

```

```

{ ----- Вывод на экран промежуточных вычислений ----- }
procedure OutputResult;
BEGIN
  ClrScr;
  WRITELN ('Вектор приближения (x1,x2) на данном шаге вычислений');
  WRITELN ('x1 = ',x[1]:9:6,' x2 = ',x[2]:9:6);
  WRITELN ('f(x1,x2) = ',f(x):9:6);
  Pausa;
END;
{ -----
                                ОСНОВНАЯ ПРОГРАММА                                ----- }
BEGIN
  ClrScr;
  WRITELN ('Введите координаты начального вектора (x1,x2)');
  FOR i := 1 TO n DO READ (x[i]);
  GradDirection;
  IF absgrad = 0 THEN
    BEGIN
      WRITELN ('Стационарная тчк. x1 = ',x[1]:9:6,' x2 = ',x[2]:9:6);
      PAUSA;
      GOTO finish;
    END;
  WRITELN ('Задайте точность нахождения точки min f(x)');
  READ (e);
  h := 0.2; e1 := e / K; OutputResult;
  REPEAT
    d := ABS(h);
    h1 := h; ScanForOneDim;
    OutputResult;
    h := h / K;
  UNTIL d < e;
  WRITELN ('Т. минимума x1 = ',x[1]:9:6,' x2 = ',x[2]:9:6);
  WRITELN ('Погрешность = ',e:9:6);
  Pausa;
finish: END.

```

4¹. Программа поиска минимума функции двух переменных методом скорейшего спуска на языке FORTRAN

```
$INCLUDE: 'EXEC.PI'
```

```

C      *****
C      ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
C      МЕТОДОМ СКОРЕЙШЕГО СПУСКА
C      *****

program MinOfFuncByCoordDescent
  INTEGER*2 system
  COMMON x(2),s(2),absgrad,K,e1,h1
  i=system('cls'C)
  WRITE (*,*) 'Введите координаты вектора (x1,x2)'
  READ (*,*) (x(1),i=1,2)
  CALL GradDirection
  IF (absgrad.LT.0.000001) THEN
    WRITE(*, '(A9,E12.6,A4,E12.6)') 'С.т. x1=',x(1),' x2=',x(2)
    PAUSE 'Нажмите клавишу ENTER для продолжения...'
    GO TO 1
  END IF
  WRITE (*,*) 'Задайте точность нахождения точки min f(x)'
  READ (*,*) e
  WRITE (*,*)
  h = 0.2
  K = 2
  e1 = e / K
  d = 1
    i=system('cls'C)
    CALL OutputResult
  DO WHILE (d.GE.e)
    d = ABS(h)
    h1 = h
    CALL ScanForOneDIM
    i=system('cls'C)
    CALL OutputResult
    h = h/K
  END DO
  WRITE (*, '(A8,E12.6,A4,E12.6)') 'Min x1=',x(1),' x2=',x(2)

```

```
WRITE (*, '(A14,E12.6)') ' Погрешность =', e
PAUSE 'Нажмите клавишу ENTER для продолжения...'
```

```
1 END
```

C

C

C

ПОДПРОГРАММЫ

```
FUNCTION f(x)
```

```
  DIMENSION x(2)
```

```
  f = (x(1)**2 + x(2) - 11)**2 + (x(1) + x(2)**2 - 7)**2
```

```
RETURN
```

```
END
```

C

```
SUBROUTINE GradDirection
```

```
  COMMON x(2),s(2),absgrad
```

```
  gradx= 4*x(1)**3 + 4*x(1)*x(2) + 2*x(2)**2 - 42*x(1) - 14
```

```
  grady= 4*x(2)**3 + 4*x(1)*x(2) + 2*x(1)**2 - 26*x(2) - 22
```

```
  absgrad = SQRT(gradx**2 + grady**2)
```

```
  s(1) = gradx / absgrad
```

```
  s(2) = grady / absgrad
```

```
RETURN
```

```
END
```

C

```
SUBROUTINE ScanForOneDIM
```

```
  COMMON x(2),s(2),absgrad,K,e1,h1
```

```
  LOGICAL a
```

```
    z = f(x)
```

```
    CALL GradDirection
```

```
    d1=1
```

```
  DO WHILE (a.OR.(d1.GE.e1))
```

```
    DO 1=1,2
```

```
      x(1) = x(1) + h1 * s(1)
```

```
    END DO
```

```
    z1 = f(x)
```

```
    d1 = ABS(h1)
```

```
    a = (z1.LT.z)
```

```
  IF (.NOT.a) h1 = -h1/K
```

```
    z = z1
```

```
END DO
```

```
RETURN
```

```
END
```

```

C ----- Вывод на экран промежуточных вычислений -----
SUBROUTINE OutputResult
COMMON x(2)
WRITE (*,*) 'Вектор приближения (x1,x2) на шаге вычислений'
WRITE (*,'(A5,E12.6,A6,E12.6)') ' x1 =',x(1),' x2 =',x(2)
WRITE (*,'(A11,E12.6)') ' f(x1,x2) =',f(x)
PAUSE 'Нажмите клавишу ENTER для продолжения...'
RETURN
END

```

5¹. Программа поиска минимума функции двух переменных методом скорейшего спуска на языке C

```

/* *****
ПОИСК ТОЧКИ МИНИМУМА ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
МЕТОДОМ СКОРЕЙШЕГО СПУСКА
***** */

#include <stdio.h>
#include <conio.h>
#include <math.h>
#define N 2
#define K 2
int i;
float absgrad,e1,h,h1,z,s[N+1],x[N+1];
/* -----
                                ПОДПРОГРАММЫ                                */
float f(float x[N+1])
{ float a,b,y; a=x[1]*x[1]+x[2]-11; b=x[1]+x[2]*x[2]-7;
  y = a*a + b*b; return y; }
/* ----- */
void graddirection(void)
{ float gradx,grady;
  gradx = 4*x[1]*x[1]*x[1]+4*x[1]*x[2]+2*x[2]*x[2]-
    42*x[1]-14;
  grady = 4*x[2]*x[2]*x[2]+4*x[1]*x[2]+2*x[1]*x[1]-
    26*x[2]-22;

```



```
char finish;
clrscr();
printf ("Введите координаты начального");
printf (" вектора (x1,x2)\n");
for (i=1; i <= N; i++) scanf ("%f",&x[i]);
graddirection();
if (!absgrad)
{ printf ("Стационарная точка x1 = %f",x[1]);
  printf (" x2 = %f",x[2]);
  pausa(); goto finish; };
printf ("Задайте точность");
printf (" нахождения точки min f(x)\n");
scanf ("%f",&e);
outputresult(); h = 0.2; e1 = e / K;
do
{ d = fabs(h);
  h1 = h; scanforonedim();
  outputresult();
  h /= K;
}
while (d >= e);
printf ("Точка минимума x1 = %f x2 = %f\n",x[1],x[2]);
printf ("Погрешность = %f\n",d);
pausa();
finish;;
}
```

Глава 1

91. 1. $\|a\|_m = 6$, $\|a\|_l = 11$, $\|a\|_h = \sqrt{61}$, $\|b\|_m = 4$, $\|b\|_l = 6$, $\|b\|_h = \sqrt{20}$,
 $d(a,b)_m = 10$, $d(a,b)_l = 13$, $d(a,b)_h = \sqrt{109}$;
 $\|c\|_m = 10$, $\|c\|_l = 22$, $\|c\|_h = 3\sqrt{20}$, $\|d\|_m = 3$, $\|d\|_l = 6$, $\|d\|_h = \sqrt{14}$,
 $d(c,d)_m = 11$, $d(c,d)_l = 24$, $d(c,d)_h = 3\sqrt{22}$.

2. а) Квадрат с вершинами в точках $A(1, 2)$, $B(-3, 2)$,
 $C(-3, -2)$, $D(1, -2)$;
б) внешность единичного круга с центром в точке $(1, 0)$;
в) граница квадрата с вершинами в точках $A(1, 0)$,
 $B(-1, 2)$, $C(-3, 0)$, $D(-1, -2)$;
г) квадрат с вершинами в точках $A(0, 3)$, $B(-6, 3)$,
 $C(-6, -3)$, $D(0, -3)$;
д) эллипс с центром в точке $(0, 0)$ и полуосями $a=2$, $b=\sqrt{3}$.
3. а) $X = \{(x_1, x_2): |x_1| < \frac{1}{4}, |x_2| < \frac{1}{2}\}$;
б) $X = \{(x_1, x_2): |x_1| > 1, |x_2| > 1\}$;
в) $X = \{(x_1, x_2): x_1 \in \mathbb{R}, 0 < x_2 < 2\}$;
г) $X = \{(x_1, x_2): |x_1| < \frac{1}{2}, -\frac{3}{4} < x_2 < \frac{1}{4}\}$;
д) $X = \{(x_1, x_2): x_1 \neq \pi k, x_2 \neq \frac{\pi}{2}(2k+1), k=0, \pm 1, \pm 2, \dots\}$.

$$4. \text{ а) } \|x\|_C = 1, \quad \|x\|_{L^2} = \frac{\sqrt{3}}{3}, \quad \|y\|_C = \frac{1}{4}, \quad \|y\|_{L^2} = \frac{\sqrt{30}}{30},$$

$$d(x,y)_C = 1, \quad d(x,y)_{L^2} = \frac{2\sqrt{30}}{15};$$

$$\text{б) } \|x\|_C = e, \quad \|x\|_{L^2} = \frac{1}{2}\sqrt{2(e^2-1)}, \quad \|y\|_C = 1, \quad \|y\|_{L^2} = \frac{\sqrt{3}}{3},$$

$$d(x,y)_C = e, \quad d(x,y)_{L^2} = \frac{1}{6}\sqrt{6[3e(e-4)+23]};$$

$$\text{в) } \|x\|_C = \ln 2, \quad \|x\|_{L^2} = \sqrt{2(1-\ln 2)}, \quad \|y\|_C = 2, \quad \|y\|_{L^2} = \sqrt{7/3},$$

$$d(x, y)_C = 2 - \ln 2, \quad d(x, y)_{L^2} = \frac{1}{6} \sqrt{6(35 + 12 \ln 2 (\ln 2 - 4))};$$

$$\Gamma) \|x\|_C = 1, \quad \|x\|_{L^2} = \frac{\sqrt{2\pi}}{2}, \quad \|y\|_C = 1, \quad \|y\|_{L^2} = \frac{\sqrt{3\pi}}{3},$$

$$d(x, y)_C = 1, \quad d(x, y)_{L^2} = \frac{1}{6} \sqrt{6(7\pi - 12)}.$$

$$\S 2. \quad 1. \text{ а) } \begin{bmatrix} 1 \\ 0,5 \end{bmatrix}; \quad \text{б) } \begin{bmatrix} 2 & 0 \\ 1 & 0 \end{bmatrix}; \quad \text{в) } \begin{bmatrix} 1 \\ -2 \\ 0 \end{bmatrix}.$$

2. а) Сходится к функции $\theta(t) \equiv 0$ ($0 \leq t \leq 2\pi$) поточечно и равномерно;

б) сходится к функции $\theta(t) \equiv 0$ ($0 \leq t \leq 1$) поточечно, но не сходится к ней равномерно;

в) сходится к функции $x(t) = |t|$ ($-\infty \leq t \leq \infty$) поточечно и равномерно;

г) сходится к разрывной функции

$$x(t) = \begin{cases} 0, & 0 < t \leq 1; \\ 1, & t = 0; \end{cases}$$

д) сходится равномерно к функции $\theta(t) \equiv 0$ ($0 \leq t \leq 1$);

е) сходится равномерно к функции $\theta(t) \equiv 0$ ($0 \leq t \leq 1$).

3. а) $\theta(t) \equiv 0$ ($0 \leq t \leq 1$); б) $\theta(t) \equiv 0$ ($0 \leq t \leq 1$).

Глава 3

- §2. 1. 2,28. 2. 2,095. 3. -1,16. 4. 1,10. 5. 3,66.
 6. -1,79. 7. -0,57. 8. 1,27. 9. -3,39. 10. 1,56.
 11. 1,57. 12. 1,94. 13. 1,64. 14. -0,375. 15. -1,15.
 16. 1,12. 17. -1,84. 18. 1,68. 19. 0,825. 20. 0,65.
 21. 1,37. 22. 5,36. 23. 1,40. 24. 0,48. 25. 1,55.

- §4. 1. $\begin{bmatrix} 2,51 \\ 3,16 \end{bmatrix}$. 2. $\begin{bmatrix} -2,51 \\ 3,16 \end{bmatrix}$. 3. $\begin{bmatrix} 0,92 \\ 1,386 \end{bmatrix}$. 4. $\begin{bmatrix} 0,83 \\ 0,56 \end{bmatrix}$.
 5. $\begin{bmatrix} -0,83 \\ -0,56 \end{bmatrix}$. 6. $\begin{bmatrix} 0,64 \\ 0,43 \end{bmatrix}$. 7. $\begin{bmatrix} 0,53 \\ -0,66 \end{bmatrix}$. 8. $\begin{bmatrix} 0,74 \\ 0,67 \end{bmatrix}$.

9. $\begin{bmatrix} -0,74 \\ -0,67 \end{bmatrix}$. 10. $\begin{bmatrix} -0,59 \\ 1,806 \end{bmatrix}$. 11. $\begin{bmatrix} 0,83 \\ -0,44 \end{bmatrix}$. 12. $\begin{bmatrix} 1,75 \\ 1,66 \end{bmatrix}$.
13. $\begin{bmatrix} 1,81 \\ 0,59 \end{bmatrix}$. 14. $\begin{bmatrix} -0,38 \\ -0,21 \end{bmatrix}$. 15. $\begin{bmatrix} 0,18 \\ 0,57 \end{bmatrix}$. 16. $\begin{bmatrix} 0,18 \\ -0,57 \end{bmatrix}$.
17. $\begin{bmatrix} 0,81 \\ 0,59 \end{bmatrix}$. 18. $\begin{bmatrix} -0,81 \\ 0,59 \end{bmatrix}$. 19. $\begin{bmatrix} 0,87 \\ 0,495 \end{bmatrix}$. 20. $\begin{bmatrix} -0,87 \\ -0,495 \end{bmatrix}$.
21. $\begin{bmatrix} 0,67 \\ 0,26 \end{bmatrix}$. 22. $\begin{bmatrix} 0,94 \\ 1,33 \end{bmatrix}$. 23. $\begin{bmatrix} 0,715 \\ 0,70 \end{bmatrix}$. 24. $\begin{bmatrix} -0,336 \\ -0,94 \end{bmatrix}$.
25. $\begin{bmatrix} 0,72 \\ 0,48 \end{bmatrix}$. 26. $\begin{bmatrix} -0,72 \\ 0,48 \end{bmatrix}$.

Глава 4

- §2. 1. $\frac{3}{2} + \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{\sin \pi(2k-1)x}{2k-1}$, $\bar{d}(f, Q_7) = 0,071$.
2. $\frac{1}{2} + \frac{4}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \pi(2k-1)x}{(2k-1)^2}$, $\bar{d}(f, Q_5) = 0,0135$.
3. $\frac{1}{2} - \frac{4}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \pi(2k-1)x}{(2k-1)^2}$, $\bar{d}(f, Q_5) = 0,0135$.
4. $\frac{\pi}{4} - \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{\cos (2k-1)x}{(2k-1)^2} + \sum_{k=1}^{\infty} (-1)^k \frac{\sin kx}{k}$, $\bar{d}(f, Q_5) = 0,235$.
5. $\frac{5\pi}{4} - \frac{10}{\pi} \sum_{k=1}^{\infty} \frac{\cos (2k-1)x}{(2k-1)^2} - \sum_{k=1}^{\infty} (-1)^k \frac{\sin kx}{k}$, $\bar{d}(f, Q_5) = 0,066$.

6. $\frac{8}{\pi} \sum_{k=1}^{\infty} \frac{\sin (2k-1)x}{2k-1}, \quad \bar{d}(f, Q_7) = 0,224.$
7. $\frac{8}{\pi} \sum_{k=1}^{\infty} \frac{\cos (2k-1)x}{(2k-1)^2}, \quad \bar{d}(f, Q_5) = 0,027.$
8. $2 \sum_{k=1}^{\infty} \frac{\sin 2kx}{k}, \quad \bar{d}(f, Q_6) = 0,305.$
9. $\pi - 2 \sum_{k=1}^{\infty} (-1)^k \frac{\sin kx}{k}, \quad \bar{d}(f, Q_7) = 0,142.$
10. $-\frac{8}{\pi} \sum_{k=1}^{\infty} \frac{\sin (2k-1)x}{2k-1}, \quad \bar{d}(f, Q_7) = 0,224.$
11. $-\frac{1}{2} - \frac{6}{\pi} \sum_{k=1}^{\infty} \frac{\sin (2k-1)x}{2k-1}, \quad \bar{d}(f, Q_7) = 0,213.$
12. $-\frac{3\pi}{4} - \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{\cos (2k-1)x}{(2k-1)^2} - \sum_{k=1}^{\infty} (-1)^k \frac{\sin kx}{k}, \quad \bar{d}(f, Q_5) = 0,117.$
13. $\frac{\pi}{4} - \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{\cos (2k-1)x}{(2k-1)^2} - 3 \sum_{k=1}^{\infty} (-1)^k \frac{\sin kx}{k}, \quad \bar{d}(f, Q_5) = 0,315.$
14. $1 - \frac{8}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \pi(2k-1)x}{(2k-1)^2}, \quad \bar{d}(f, Q_5) = 0,0135.$

$$15. \quad 3 - \frac{8}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \frac{\pi(2k-1)x}{2}}{(2k-1)^2}, \quad \bar{d}(f, Q_5) = 0,0051.$$

$$16. \quad 1 + \frac{8}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \frac{\pi(2k-1)x}{2}}{(2k-1)^2}, \quad \bar{d}(f, Q_5) = 0,0135.$$

$$17. \quad -\frac{5\pi}{4} + \frac{10}{\pi} \sum_{k=1}^{\infty} \frac{\cos (2k-1)x}{(2k-1)^2} + \sum_{k=1}^{\infty} (-1)^k \frac{\sin kx}{k}, \quad \bar{d}(f, Q_5) = 0,066.$$

$$18. \quad \frac{1}{2} - \frac{4}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \frac{\pi(2k-1)x}{2}}{(2k-1)^2} - \frac{2}{\pi} \sum_{k=1}^{\infty} (-1)^k \frac{\sin \frac{\pi kx}{2}}{k}, \quad \bar{d}(f, Q_5) = 0,235.$$

$$19. \quad 1 - \frac{8}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \frac{\pi(2k-1)x}{4}}{(2k-1)^2} - \frac{12}{\pi} \sum_{k=1}^{\infty} (-1)^k \frac{\sin \frac{\pi kx}{4}}{k}, \quad \bar{d}(f, Q_5) = 0,315.$$

$$20. \quad \frac{1}{2} - \frac{4}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \frac{\pi(2k-1)x}{2}}{(2k-1)^2} + \frac{2}{\pi} \sum_{k=1}^{\infty} (-1)^k \frac{\sin \frac{\pi kx}{2}}{k}, \quad \bar{d}(f, Q_4) = 0,26.$$

$$21. \quad 1 + \frac{8}{\pi} \sum_{k=1}^{\infty} \frac{\sin \pi(2k-1)x}{2k-1}, \quad \bar{d}(f, Q_7) = 0,201.$$

$$22. \quad -1 - \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{\sin \frac{\pi(2k-1)x}{2}}{2k-1}, \quad \bar{d}(f, Q_7) = 0,452.$$

$$23. \quad \frac{5}{4} - \frac{2}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \pi(2k-1)x}{(2k-1)^2} - \frac{1}{\pi} \sum_{k=1}^{\infty} (-1)^k \frac{\sin \pi kx}{k},$$

$$\bar{a}(f, Q_4) = 0,0822.$$

$$24. \quad \frac{3}{4} + \frac{2}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \pi(2k-1)x}{(2k-1)^2} - \frac{1}{\pi} \sum_{k=1}^{\infty} (-1)^k \frac{\sin \pi kx}{k},$$

$$\bar{a}(f, Q_4) = 0,130.$$

$$25. \quad \frac{1}{2} + \frac{4}{\pi^2} \sum_{k=1}^{\infty} \frac{\cos \frac{\pi(2k-1)x}{2}}{(2k-1)^2} + \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{\sin \frac{\pi kx}{2}}{k},$$

$$\bar{a}(f, Q_5) = 0,608.$$

$$\S 3. \quad 1. \quad Q_2(x) = (15x^2 + 32x - 116)/64, \quad \bar{a}(f, Q_2) = 0,044.$$

$$2. \quad Q_2(x) = -(75x^2 + 32x + 60)/64, \quad \bar{a}(f, Q_2) = 0,122.$$

$$3. \quad Q_2(x) = (15x^2 - 30x + 2)/16, \quad \bar{a}(f, Q_2) = 0,125.$$

$$4. \quad Q_2(x) = (-15x^2 + 96x - 12)/128, \quad \bar{a}(f, Q_2) = 0,040.$$

$$5. \quad Q_2(x) = -(45x^2 + 32x + 36)/64, \quad \bar{a}(f, Q_2) = 0,118.$$

$$6. \quad Q_2(x) = (45x^2 + 32x + 36)/64, \quad \bar{a}(f, Q_2) = 0,118.$$

$$7. \quad Q_2(x) = (15x^2 + 96x - 4)/64, \quad \bar{a}(f, Q_2) = 0,240.$$

$$8. \quad Q_2(x) = (-15x^2 + 48x - 27)/96, \quad \bar{a}(f, Q_2) = 0,088.$$

$$9. \quad Q_2(x) = (15x^2 - 14x + 34)/32, \quad \bar{a}(f, Q_2) = 0,028.$$

$$10. \quad Q_4(x) = (-105x^4 + 210x^2 + 128x + 15)/256, \quad \bar{a}(f, Q_4) = 0,044.$$

$$11. \quad Q_2(x) = (-15x^2 - 16x + 29)/32, \quad \bar{a}(f, Q_2) = 0,044.$$

$$12. \quad Q_2(x) = (75x^2 + 32x - 60)/64, \quad \bar{a}(f, Q_2) = 0,122.$$

$$13. \quad Q_2(x) = (-15x^2 - 32x + 116)/64, \quad \bar{a}(f, Q_2) = 0,044.$$

14. $Q_2(x) = (15x^2 - 96x + 12)/128$, $\bar{d}(f, Q_2) = 0,040$.
15. $Q_4(x) = (105x^4 - 210x^2 - 128x - 15)/256$, $\bar{d}(f, Q_4) = 0,044$.
16. $Q_5(x) = (693x^5 - 1050x^3 + 525x + 128)/256$, $\bar{d}(f, Q_5) = 0,221$.
17. $Q_4(x) = (105x^4 - 840x^2 + 1808)/924$, $\bar{d}(f, Q_4) = 0,062$.
18. $Q_4(x) = (945x^4 - 3780x^3 + 5500x^2 - 3480x + 832)/924$, $\bar{d}(f, Q_4) = 0,091$.
19. $Q_4(x) = (315x^4 - 630x^2 + 339)/384$, $\bar{d}(f, Q_4) = 0,062$.
20. $Q_4(x) = (-105x^4 + 840x^2 + 240)/512$, $\bar{d}(f, Q_4) = 0,062$.
21. $Q_4(x) = (-315x^4 + 630x^2 - 339)/384$, $\bar{d}(f, Q_4) = 0,062$.
22. $Q_5(x) = (693x^5 - 4200x^3 + 5760x + 6736)/4096$, $\bar{d}(f, Q_5) = 0,022$.
23. $Q_4(x) = (-105x^4 + 210x^2 + 143)/128$, $\bar{d}(f, Q_4) = 0,024$.
24. $Q_5(x) = (693x^5 - 1050x^3 + 525x)/128$, $\bar{d}(f, Q_5) = 0,312$.
25. $Q_4(x) = (-105x^4 + 210x^2 + 15)/128$, $\bar{d}(f, Q_4) = 0,062$.

§4.

Коэффициенты многочленов наилучшего приближения

 $Q_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$ и погрешности $\bar{d}(f, Q_n)$:

Номер вар.	$n \backslash a_i$	a_0	a_1	a_2	a_3	$\bar{d}(f, Q_n)$
1	2	0,8200	0,31857	-0,0214	-	0,0193
	3	0,5401	0,7118	-0,1714	0,016663	0,0068
2	2	0,56	0,59857	-0,07143	-	0,03146
	3	0,07013	1,287	-0,334	0,0292	0,00503
3	2	0,736	-0,135	0,003	-	0,347
	3	-1,7423	2,0582	-0,5872	0,04917	0,2231
4	2	8,8	-1,2986	0,02143	-	0,0521
	3	7,54	0,4715	-0,6536	0,075	0,0465

Номер вар.	$n \backslash a_i$	a_0	a_1	a_2	a_3	$\bar{a}(f, q_n)$
5	2	10,36	-2,519	0,2214	-	0,0329
	3	12,039	-4,8776	1,121	-0,1	0,0116
6	2	8,56	-1,461	0,0786	-	0,0058
	3	8,84	-1,8542	0,2284	-0,0166	0,0020
7	2	8,32	-1,2243	0,03571	-	0,0098
	3	8,04	-0,8308	-0,1143	0,01667	0,0082
8	2	0,392	0,1834	-0,0186	-	0,0122
	3	0,308	0,3014	-0,0636	0,005	0,0043
9	2	0,64	0,4821	-0,018	-	0,0082
	3	0,64	0,4821	-0,018	0	0,0082
10	2	0,61	0,5214	-0,0286	-	0,0056
	3	0,61	0,5214	-0,0286	0	0,0056
11	2	6,53	-1,4514	0,1286	-	0,0392
	3	5,3399	0,22033	-0,50896	0,07084	0,0222
12	2	6,53	-1,78	0,2	-	0,0595
	3	8,1396	-4,04101	1,0623	-0,09581	0,0373
13	2	-0,98	2,4986	0,3786	-	0,00396
	3	-1,1203	2,6957	0,3034	0,00835	0,0038
14	2	2,496	0,6093	-0,0707	-	0,0122
	3	2,074	1,1992	-0,2957	0,025	0,0027
15	2	0,388	-0,1746	0,02143	-	0,15896
	3	0,57	-0,4302	0,1189	-0,01083	0,0635
16	2	0,3866	-0,1665	0,01986	-	0,1033
	3	0,5154	-0,3474	0,08884	-0,00766	0,0255
17	2	0,108	0,104	-0,01	-	0,00893
	3	0,08	0,1433	-0,025	0,001667	0,00000

Номер вар.	$n \backslash a_i$	a_0	a_1	a_2	a_3	$\bar{\sigma}(f, q_n)$
18	2	4,104	0,5882	0,1159	-	0,00157
	3	3,989	0,7496	0,0543	0,006835	0,000044
19	2	-1,238	2,537	-0,227	-	0,02277
	3	-2,176	3,8544	-0,7296	0,055826	0,00417
20	2	0,138	0,2173	-0,0207	-	0,01046
	3	0,082	0,2959	-0,05071	0,00333	0,003697
21	2	1,254	0,2977	-0,03429	-	0,01309
	3	1,03	0,6124	-0,1543	0,01333	0,00301
22	2	1,7012	-0,8952	0,1158	-	0,22478
	3	2,674	-2,2618	0,6369	-0,05791	0,07088
23	2	1,524	-0,6696	0,08643	-	0,14563
	3	2,2659	-1,7117	0,4839	-0,04416	0,04269
24	2	2,288	-0,4137	0,4443	-	0,01488
	3	1,2938	0,9829	-0,08832	0,05918	0,001341
25	2	1,252	0,8463	-0,07571	-	0,01005
	3	0,958	1,2593	-0,2332	0,0175	0,001952

§5.

Эмпирическая формула $y = Q(x, \alpha, \beta)$, где α и β — наилучшие значения параметров, полученные методом наименьших квадратов:

Номер вар.	Эмпирическая формула	Номер вар.	Эмпирическая формула	Номер вар.	Эмпирическая формула
1	$0,481 \ln x + 1,084$	2	$1,98 - \frac{0,89}{x}$	3	$\frac{1}{3,286x - 5,66}$
4	$9,978 \cdot 0,786^x$	5	$-3,041 \ln x + 8,15$	6	$8,852 \cdot 0,818^x$
7	$-1,01x + 8,07$	8	$\frac{x}{1,015x + 0,8216}$	9	$0,375x + 0,765$

Номер вар.	Эмпирическая формула	Номер вар.	Эмпирическая формула	Номер вар.	Эмпирическая формула
10	$1,0936x^{0,513}$	11	$\frac{1}{0,0535x+0,1391}$	12	$1,93 + \frac{3,22}{x}$
13	$1,924x^{1,489}$	14	$4 - \frac{0,998}{x}$	15	$0,23x^{-1,111}$
16	$\frac{1}{4,948x - 0,8892}$	17	$\frac{x}{2,028x+3,015}$	18	$4 \cdot 1,2^x$
19	$3 \ln x + 1$	20	$\frac{x}{0,999x+2,01}$	21	$1,998 - \frac{0,5}{x}$
22	$\frac{1}{3,989x-2,96}$	23	$\frac{x}{4,039x-3,069}$	24	$1,498 \cdot 1,5^x$
25	$1,002 \ln x + 1,999$				

Глава 9

92. 1. $\mathbf{x}^* = \left(1, \frac{1}{2} \right)$, $D = \{ (x_1, x_2) : x_1 > 0, x_2 > \frac{1}{8x_1} \}$.
2. $\mathbf{x}^* = (-2, 0)$, $D = \{ (x_1, x_2) : x_1 > -4, x_2^2 < 4+x \}$.
3. $\mathbf{x}^* = (0, 3)$, $D = \mathbb{R}^2$.
4. $\mathbf{x}^* = (2, -2)$, $D = \mathbb{R}^2$.
5. $\mathbf{x}^* = (-1, 1)$, $D = \mathbb{R}^2$.
6. $\mathbf{x}^* = (5, 6)$, $D = \{ (x_1, x_2) : x_1 > 3, x_2 \in \mathbb{R} \}$.
7. $\mathbf{x}^* = (4, 1)$, $D = \mathbb{R}^2$.
8. $\mathbf{x}^* = (1, 1)$, $D = \{ (x_1, x_2) : x_1 \in \mathbb{R}, x_2 > 0 \}$.
9. $\mathbf{x}^* = (0, 3)$, $D = \mathbb{R}^2$.
10. $\mathbf{x} = (2, 2)$, $D = \{ (x_1, x_2) : x_1 > 0, x_2 > 0 \}$.
11. $\mathbf{x}^* = (0, 0)$, $D = \left\{ (x_1, x_2) : x_1 > -\frac{5}{6}, x_2^2 < x_1 - 6x_1^2 + 5 \right\}$.
12. $\mathbf{x}^* = (1, -2)$, $D = \mathbb{R}^2$.
13. $\mathbf{x}^* = (0, 3)$, $D = \mathbb{R}^2$.

$$14. \mathbf{x}^* = \left[-\frac{2}{3}, \frac{5}{3} \right], \quad D = \mathbb{R}^2.$$

$$15. \mathbf{x}^* = (5, 2), \quad D = \{(x_1, x_2): x_1 > 0, \quad x_1 x_2 < 10 \sqrt[3]{4}\}.$$

$$16. \mathbf{x}^* = (-1, -2), \quad D = \mathbb{R}^2.$$

$$17. \mathbf{x}^* = (1, 3), \quad D = \{(x_1, x_2): x_1 > 0, \quad x_2 > 0\}.$$

$$18. \mathbf{x}^* = (5, 5), \quad D = \{(x_1, x_2): x_1 > 0, \quad x_1 x_2 > \frac{25}{4}\}.$$

$$19. \mathbf{x}^* = (0, 0), \quad D = \mathbb{R}^2.$$

$$20. \mathbf{x}^* = (0, 0), \quad D = \mathbb{R}^2.$$

$$21. \mathbf{x}^* = \left[\frac{\sqrt{6}}{2}, \frac{1}{2} \right], \quad D = \{(x_1, x_2): x_1 > 0, \quad 13x_1^2 - 5x_2 - 5 > 0\}.$$

$$22. \mathbf{x}^* = \left[-\frac{\sqrt{6}}{2}, \frac{1}{2} \right], \quad D = \{(x_1, x_2): x_1 < 0, \quad 13x_1^2 - 5x_2 - 5 > 0\}.$$

$$23. \mathbf{x}^* = (1, 1), \quad D = \{(x_1, x_2): x_1 > 0, \quad 2x_1^2 - x_2 > 0\}.$$

$$24. \mathbf{x}^* = (-1, 1), \quad D = \{(x_1, x_2): x_1 < 0, \quad 2x_1^2 - x_2 > 0\}.$$

$$25. \mathbf{x}^* = (-2, 1), \quad D = \mathbb{R}^2.$$

- §3. 1. $\mathbf{x}^* = 0, \quad -2 < x < \infty.$ 2. $\mathbf{x}^* = -1, \quad -3 < x < \infty.$
 3. $\mathbf{x}^* = 4, \quad 2 < x < \infty.$ 4. $\mathbf{x}^* = -3, \quad -\infty < x < \infty.$
 5. $\mathbf{x}^* = 2, \quad 0 < x < \infty.$ 6. $\mathbf{x}^* = 1, \quad 0 < x < \infty.$
 7. $\mathbf{x}^* = 1, \quad -1 < x < \infty.$ 8. $\mathbf{x}^* = -1, \quad -\infty < x < 1.$
 9. $\mathbf{x}^* = 2, \quad 0 < x < \infty.$ 10. $\mathbf{x}^* = 2, \quad 0 < x < \infty.$
 11. $\mathbf{x}^* = 0, \quad -\infty < x < 2.$ 12. $\mathbf{x}^* = -2, \quad -\infty < x < 0.$
 13. $\mathbf{x}^* = 1, \quad -\infty < x < \infty.$ 14. $\mathbf{x}^* = -1, \quad -2 < x < \infty.$
 15. $\mathbf{x}^* = 0, \quad -\infty < x < \infty.$ 16. $\mathbf{x}^* = e, \quad 0 < x < \infty.$
 17. $\mathbf{x}^* = -3, \quad -\infty < x < -2.$ 18. $\mathbf{x}^* = 1, \quad 0 < x < \infty.$
 19. $\mathbf{x}^* = 0, \quad -\infty < x < 1.$ 20. $\mathbf{x}^* = 1, \quad -\infty < x < 2.$
 21. $\mathbf{x}^* = -1, \quad -\sqrt{2} < x < 1.$ 22. $\mathbf{x}^* = 1, \quad 0 < x < \infty.$
 23. $\mathbf{x}^* = -1, \quad -\infty < x < 0.$ 24. $\mathbf{x}^* = 2, \quad 1 < x < \infty.$
 25. $\mathbf{x}^* = 4, \quad -\infty < x < \infty.$

ЛИТЕРАТУРА

1. Быхвалов Н.С., Жидков Н.П., Кобальков Г.М. Численные методы. - М.: Наука, 1987.
2. Башмакова Е.С., Виттенберг И.М., Либеров А.Б., Панков А.А. Справочник. Программирование микроЕМ на языке Бейсик. - М.: Радио и связь, 1991.
3. Демидович Б.П., Марон И.А. Основы вычислительной математики. - М.: Наука, 1963.
4. Демидович Б.П., Марон И.А., Шувалова Э.З. Численные методы анализа. - М.: Наука, 1963.
5. Волков Е.А. Численные методы. - М.: Наука, 1987.
6. Вулик Б.З. Введение в функциональный анализ. - М.: Наука, 1967.
7. Менсен К., Вирт Н. ПАСКАЛЬ. Руководство для пользователя. - М.: Финансы и статистика, 1989.
8. Катцан Г. Язык Фортран 77. - М.: Мир, 1982.
9. Керниган Б., Ритчи Д. Язык программирования Си. - М.: Финансы и статистика, 1992.
10. Марчук Г.М. Методы вычислительной математики. - М.: Наука, 1980.
11. Ортега Дж., Пул У. Введение в численные методы решения дифференциальных уравнений. - М.: Наука, 1986.
12. Осипов С.В., Потемкин В.Г., Симоненков О.С. MS-DOS 5.0. QBasic. - М.: Малин, 1992.
13. Реклейтис Г., Рейвиндран А., Рэгсдел К. Оптимизация в технике. Кн. 1. - М.: Мир, 1986.
14. Сборник задач по методам вычислений. Под ред. Монастырского П.И. - М.: Наука, 1994.
15. Соловьев П.В. FORTRAN для персонального компьютера (справочное пособие). - М.: Арист, 1991.
16. Сухарев А.Г., Тимохов А.В., Федоров В.В. Курс методов оптимизации. - М.: Наука, 1986.
17. Треногин В.А. Функциональный анализ. - М.: Наука, 1980.
18. Турчак Л.И. Основы численных методов. - М.: Наука, 1987.
19. Уэйт М., Прета С., Мартин Д. Язык Си. Руководство для начинающих. - М.: Мир, 1988.

ОГЛАВЛЕНИЕ

Предисловие.....	3
Глава 1. Понятие линейного нормированного пространства.....	5
§1. Основные определения. Примеры линейных нормиро- ванных пространств.....	5
§2. Сходимость последовательностей в линейных норми- рованных пространствах.....	14
1 ⁰ . Сходимость последовательностей n -мерных векто- ров и матриц.....	14
2 ⁰ . Сходимость последовательностей непрерывных функций.....	16
Глава 2. Методы численного решения систем линейных алгебраических уравнений.....	21
§1. Метод Гаусса.....	21
§2. Метод итераций.....	29
Глава 3. Методы численного решения уравнений и систем нелинейных уравнений.....	36
§1. Отделение корней.....	36
§2. Метод половинного деления для уравнения $f(x)=0$	37
§3. Метод итераций для одного уравнения с одним неизвестным.....	40
§4. Метод итераций для системы двух нелинейных уравнений.....	45
Глава 4. Среднеквадратичное приближение функций.....	53
§1. Интегральное среднеквадратичное приближение функций обобщенными многочленами.....	53
§2. Среднеквадратичное приближение функций тригонометрическими многочленами.....	56
§3. Среднеквадратичное приближение функций алгебраическими многочленами Лежандра.....	64
§4. Точечное среднеквадратичное приближение функций ортгоналными многочленами. Ортогональные мно- гочлены Чебышева.....	73
§5. Метод наименьших квадратов. Эмпирические формулы....	81

Глава 5. Интерполирование функций	88
§1. Интерполяционная формула Лагранжа.....	88
§2. Интерполирование функций кубическими сплайнами.....	93
Глава 6. Численное дифференцирование.....	99
§1. Вычисление производной по ее определению.....	99
§2. Конечно-разностные аппроксимации производных.....	100
§3. Использование интерполяционных многочленов Лагранжа для формул численного дифференцирования.....	102
Глава 7. Численное интегрирование.....	107
§1. Квадратурные формулы прямоугольников, трапеций и Симпсона.....	107
§2. Квадратурные формулы Гаусса.....	113
§3. Приближенное вычисление несобственных интегралов интегралов с бесконечными пределами.....	117
§4. Приближенное вычисление несобственных интегралов интегралов от функций с бесконечным разрывом.....	123
Глава 8. Численное решение обыкновенных дифференциальных уравнений.....	129
§1. Численное решение дифференциальных уравнений первого порядка.....	129
1 ⁰ . Понятие о численном решении задачи Коши.....	129
2 ⁰ . Метод Эйлера.....	131
3 ⁰ . Методы Рунге - Кутты.....	133
§2. Численное решение систем дифференциальных уравнений первого порядка.....	139
§3. Численное решение дифференциальных уравнений и систем дифференциальных уравнений высших порядков.....	145
Глава 9. Численные методы безусловной оптимизации.....	150
§1. Необходимые и достаточные условия экстремума в задачах безусловной оптимизации.....	150
§2. Выпуклые множества и выпуклые функции.....	154
§3. Численные методы поиска минимума функции одной переменной.....	158
1 ⁰ . Унимодальные функции.....	158

2 ⁰ . Схема сужения промежутка унимодальности функции...	159
3 ⁰ . Метод половинного деления.....	161
4 ⁰ . Метод золотого сечения.....	164
5 ⁰ . Метод сканирования.....	167
§4. Численные методы поиска минимума функции	
нескольких переменных.....	171
1 ⁰ . Общая схема методов спуска.....	171
2 ⁰ . Метод покоординатного опускания.....	174
3 ⁰ . Метод скорейшего спуска.....	176
4 ⁰ . Применение методов спуска к решению систем нелинейных уравнений.....	179
Приложения.....	181
Основные блоки и логические управляющие структуры.....	181
Приложение к главе 2.....	184
К §1. Решение системы линейных алгебраических уравне- ний методом Гаусса с выбором главного элемента (по столбцу)	
1 ⁰ . Блок-схема.....	184
2 ⁰ . Программа на языке BASIC.....	186
3 ⁰ . Программа на языке PASCAL.....	189
4 ⁰ . Программа на языке FORTRAN.....	192
5 ⁰ . Программа на языке C.....	195
6 ⁰ . Программа решения системы линейных алгебраи- ческих уравнений методом Жордана - Гаусса на языке QUICKBASIC.....	197
К §2. Решение системы линейных алгебраических урав- нений методом итераций	
1 ⁰ . Блок-схема.....	202
2 ⁰ . Программа на языке BASIC.....	204
3 ⁰ . Программа на языке PASCAL.....	207
4 ⁰ . Программа на языке FORTRAN.....	212
5 ⁰ . Программа на языке C.....	216
Приложение к главе 3.....	220
К §2. Решение уравнения $f(x)=0$ методом половинного деления	
1 ⁰ . Блок-схема.....	220
2 ⁰ . Программа на языке BASIC.....	220
3 ⁰ . Программа на языке PASCAL.....	221

4 ⁰ . Программа на языке FORTRAN.....	222
5 ⁰ . Программа на языке C.....	224
К §3. Решение уравнения $f(x)=0$ методом итераций	
1 ⁰ . Блок-схема.....	225
2 ⁰ . Программа на языке BASIC.....	226
3 ⁰ . Программа на языке PASCAL.....	226
4 ⁰ . Программа на языке FORTRAN.....	227
5 ⁰ . Программа на языке C.....	228
К §4. Решение системы нелинейных уравнений методом итераций	
1 ⁰ . Блок-схема.....	229
2 ⁰ . Программа на языке BASIC.....	230
3 ⁰ . Программа на языке PASCAL.....	231
4 ⁰ . Программа на языке FORTRAN.....	232
5 ⁰ . Программа на языке C.....	233
Приложение к главе 4.....	235
К §2. Построение тригонометрического многочлена, аппроксимирующего заданную функцию	
1 ⁰ . Блок-схема.....	235
2 ⁰ . Программа на языке BASIC.....	236
3 ⁰ . Программа на языке PASCAL.....	236
4 ⁰ . Программа на языке FORTRAN.....	238
5 ⁰ . Программа на языке C.....	239
К §4. Вычисление ортогональных многочленов Чебышева на заданном множестве точек	
1 ⁰ . Блок-схема.....	240
2 ⁰ . Программа на языке BASIC.....	241
3 ⁰ . Программа на языке PASCAL.....	243
4 ⁰ . Программа на языке FORTRAN.....	245
5 ⁰ . Программа на языке C.....	247
Вычисление многочленов наилучшего ординеквадратичного приближения функций ортогональными многочленами	
1 ¹ . Алгоритм вычисления.....	248
2 ¹ . Программа на языке BASIC.....	249
3 ¹ . Программа на языке PASCAL.....	251
4 ¹ . Программа на языке FORTRAN.....	254
5 ¹ . Программа на языке C.....	256

К §5. Определение параметров эмпирической формулы с двумя параметрами методом наименьших квадратов	
1 ⁰ . Блок-схема.....	259
2 ⁰ . Программа на языке BASIC.....	260
3 ⁰ . Программа на языке PASCAL.....	261
4 ⁰ . Программа на языке FORTRAN.....	264
5 ⁰ . Программа на языке C.....	267
6 ⁰ . Программа на языке QUICKBASIC.....	268
Приложение к главе 5.....	273
К §1. Построение интерполяционного многочлена Лагранжа	
1 ⁰ . Блок-схема.....	273
2 ⁰ . Программа на языке BASIC.....	274
3 ⁰ . Программа на языке PASCAL.....	274
4 ⁰ . Программа на языке FORTRAN.....	276
5 ⁰ . Программа на языке C.....	277
К §2. Построение кубического сплайна	
1 ⁰ . Блок-схема.....	277
2 ⁰ . Программа на языке BASIC.....	279
3 ⁰ . Программа на языке PASCAL.....	280
4 ⁰ . Программа на языке FORTRAN.....	282
5 ⁰ . Программа на языке C.....	283
Приложение к главе 6.....	285
К §1. Вычисление производной по ее определению	
1 ⁰ . Блок-схема.....	285
2 ⁰ . Программа на языке BASIC.....	285
3 ⁰ . Программа на языке PASCAL.....	286
4 ⁰ . Программа на языке FORTRAN.....	287
5 ⁰ . Программа на языке C.....	288
К §3. Вычисление производных первого и второго порядков с одинаковым порядком аппроксимации по шагу h	
1 ⁰ . Блок-схема.....	289
2 ⁰ . Программа на языке BASIC.....	290
3 ⁰ . Программа на языке PASCAL.....	291
4 ⁰ . Программа на языке FORTRAN.....	293
5 ⁰ . Программа на языке C.....	294
Приложение к главе 7.....	296
К §1. Вычисление определенного интеграла по квадратурным формулам прямоугольников, трапеций и Симпсона	

1 ⁰ . Блок-схема.....	296
2 ⁰ . Программа на языке BASIC.....	296
3 ⁰ . Программа на языке PASCAL.....	297
4 ⁰ . Программа на языке FORTRAN.....	298
5 ⁰ . Программа на языке C.....	299
Вычисление определенного интеграла методом	
двойного пересчета по формуле Симпсона	
1 ¹ . Блок-схема.....	300
2 ¹ . Программа на языке BASIC.....	301
3 ¹ . Программа на языке PASCAL.....	302
4 ¹ . Программа на языке FORTRAN.....	303
5 ¹ . Программа на языке C.....	304
К §2. Вычисление определенного интеграла методом двой-	
ного пересчета по формуле Гаусса с тремя узлами	
1 ⁰ . Блок-схема.....	306
2 ⁰ . Программа на языке BASIC.....	307
3 ⁰ . Программа на языке PASCAL.....	308
4 ⁰ . Программа на языке FORTRAN.....	309
5 ⁰ . Программа на языке C.....	310
Приложение к главе 8.....	312
К §1. Решение задачи Коши для дифференциального уравне-	
ния первого порядка методами Эйлера, Эйлера - Коши	
и Рунге - Кутта	
1 ⁰ . Блок-схема.....	312
2 ⁰ . Программа на языке BASIC.....	313
3 ⁰ . Программа на языке PASCAL.....	314
4 ⁰ . Программа на языке FORTRAN.....	315
5 ⁰ . Программа на языке C.....	317
К §2. Решение задачи Коши для системы дифференциальных	
уравнений первого порядка методом Рунге - Кутта	
1 ⁰ . Блок-схема.....	318
2 ⁰ . Программа на языке BASIC.....	319
3 ⁰ . Программа на языке PASCAL.....	320
4 ⁰ . Программа на языке FORTRAN.....	322
5 ⁰ . Программа на языке C.....	324
6 ⁰ . Программа на языке QUICKBASIC.....	325

Приложение к главе 9.....	328
К §3. Поиск минимума функции одной переменной методом половинного деления	
1 ⁰ . Блок-схема.....	328
2 ⁰ . Программа на языке BASIC.....	329
3 ⁰ . Программа на языке PASCAL.....	330
4 ⁰ . Программа на языке FORTRAN.....	331
5 ⁰ . Программа на языке C.....	332
Поиск минимума функции одной переменной методом золотого сечения	
1 ¹ . Блок-схема.....	334
2 ¹ . Программа на языке BASIC.....	334
3 ¹ . Программа на языке PASCAL.....	335
4 ¹ . Программа на языке FORTRAN.....	337
5 ¹ . Программа на языке C.....	339
Поиск минимума функции одной переменной методом сканирования	
1 ² . Блок-схема.....	341
2 ² . Программа на языке BASIC.....	341
3 ² . Программа на языке PASCAL.....	342
4 ² . Программа на языке FORTRAN.....	344
5 ² . Программа на языке C.....	345
К §4. Поиск минимума функции двух переменных методом покоординатного опуска	
1 ⁰ . Блок-схема.....	346
2 ⁰ . Программа на языке BASIC.....	348
3 ⁰ . Программа на языке PASCAL.....	349
4 ⁰ . Программа на языке FORTRAN.....	351
5 ⁰ . Программа на языке C.....	353
Поиск минимума функции двух переменных методом скорейшего спуска	
1 ¹ . Блок-схема.....	355
2 ¹ . Программа на языке BASIC.....	355
3 ¹ . Программа на языке PASCAL.....	357
4 ¹ . Программа на языке FORTRAN.....	360
5 ¹ . Программа на языке C.....	362
Ответы к упражнениям.....	365
Литература	376

Учебное издание

**Ракитин Валентин Иванович
Первушин Виталий Евгеньевич**

**ПРАКТИЧЕСКОЕ РУКОВОДСТВО
ПО МЕТОДАМ ВЫЧИСЛЕНИЙ
С ПРИЛОЖЕНИЕМ ПРОГРАММ
ДЛЯ ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРОВ**

**Редактор А. М. Суходский
Художественный редактор Т. А. Колешкова
Технический редактор В. М. Романова
Оригинал-макет книги выполнен В. И. Ракитиным**

ЛР № 010146 от 25.12.96. Изд. № ФМ — 161. Сдано в набор и подписано в печать 31. 10. 97. Формат 60 × 88/16. Бум. газетн. Печать офсетная. Объем 23,52 усл. печ. л. 23, 77 усл. кр.-отт. 20,22 уч. изд. л. Тираж 10000 экз. Зак. № 469

Издательство «Высшая школа», 101430, Москва, ГСП-4, Неглинная ул., д. 29/14.

Отпечатано с диапозитивов издательства в ОАО «Оригинал», 101898, Москва, Хохловский пер., д. 7.